Document UM-00070-F

User Manual
for

DT3362 SERIES

HIGH PERFORMANCE

A/D SUBSYSTEMS

For Q-bus Processors

Data Translation, Inc.
100 Locke Drive
Marlborough, MA 01752-1192

Fourth Edition
July, 1986

July, 1986 Printing

Information furnished by Data Translation, Inc. is believed to be accurate and reliable; however, no responsibility is assumed by Data Translation, Inc. for its use; nor for any infringements of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent rights of Data Translation, Inc.

CONTENTS

PREFACE

CHAPTER 1 INTRODUCTION

1.1 MODEL DESCRIPTIONS 1-2
1.2 COMPATIBLE SCREW TERMINAL PANELS 1-6
1.2.1 DT701 SERIES 1-6
1.2.2 DT6700 SERIES 1-6
1.2.3 DT709 SERIES 1-6
1.2.4 DT756 SERIES 1-7

CHAPTER 2 SPECIFICATIONS

2.1 INTRODUCTION 2-1
2.2 DT3362-16SE/8DI AND DT3362-64SE/32DI 2-2
2.3 DT3362-16SE/8DI-PGH AND DT3362-64SE/32DI-PGH . . . 2-5
2.4 DT3362-F-16SE AND DT3362-F-64SE 2-8
2.5 DT3362-F-8DI AND DT3362-F-32DI 2-11
2.6 DT3362-G-16SE AND DT3362-G-64SE 2-14
2.7 DT3362-G-8DI AND DT3362-G-32DI 2-17
2.8 DT3362-H-16SE AND DT3362-H-64SE 2-20
2.9 DT3362-H-8DI AND DT3362-H-32DI 2-23
2.10 DT3362-H-16SE-PGH AND DT3362-H-64SE-PGH 2-25
2.11 DT3362-H-8DI-PGH AND DT3362-H-32DI-PGH 2-28
2.12 DT3377 2-31
2.13 DT3368-4SE AND DT3368-12SE 2-33
2.14 EXTERNAL TRIGGER 2-36
2.15 INTERFACE CHARACTERISTICS 2-36
2.16 POWER REQUIREMENTS 2-37
2.17 PHYSICAL/ENVIRONMENTAL 2-37
2.18 CONNECTORS AND CABLES 2-37
2.19 COMPATIBLE SCREW TERMINAL PANELS 2-38

CHAPTER 3 UNPACKING AND CONFIGURATION

3.1 INTRODUCTION 3-1
3.2 UNPACKING 3-1
3.2.1 NO APPARENT DAMAGE 3-1
3.2.2 VISIBLE DAMAGE 3-2
3.3 CONFIGURATION 3-3
3.3.1 DEVICE BASE ADDRESS SELECTION 3-3
3.3.2 DEVICE INTERRUPT VECTOR ADDRESS SELECTION . . . 3-4
3.4 A/D CONFIGURATION JUMPERING 3-5
3.4.1 INPUT MODE 3-5
3.4.2 A/D INPUT RANGE 3-7
3.4.3 SELECTION OF OUTPUT CODING 3-10
3.4.3.1 Input Gain And A/D Range 3-12
3.4.4 JUMPER CONFIGURATIONS ON 22-BIT DMA ADDRESS BOARD 3-13
3.4.4.1 Selection Of DMA Wrap Mode 3-13

3.4.4.2 Triggers 3-14
3.4.4.3 Reference Jumpers 3-15

CHAPTER 4 SYSTEM INTERCONNECTIONS

4.1 INTRODUCTION 4-1
4.2 ANALOG INPUT CONNECTIONS 4-1
4.2.1 J3 AND J2 CONNECTIONS ON DT3362- PREFIXED MODELS 4-1
4.2.2 J3 CONNECTION ON DT3377 4-3
4.2.3 J3 AND J2 CONNECTIONS ON DT3368- PREFIXED MODELS 4-4
4.3 EXTERNAL PORT CONNECTION 4-5
4.4 EXTERNAL TRIGGER INPUT 4-6
4.5 CONNECTIONS TO DATA ACQUISITION ACCESSORIES . . . 4-6
4.6 RECOMMENDED ANALOG INPUT CONNECTION SCHEMES . . . 4-9
4.6.1 SINGLE-ENDED INPUTS 4-11
4.6.2 PSEUDO-DIFFERENTIAL INPUTS 4-13
4.6.3 DIFFERENTIAL INPUTS 4-15
4.7 CONNECTION GUIDELINES 4-17
4.7.1 TWISTED PAIR INPUT LINES 4-17
4.7.2 SHIELDED INPUT LINES 4-17
4.7.3 INPUT SETTLING WITH HIGH SOURCE IMPEDANCE . . 4-17

CHAPTER 5 ARCHITECTURE

5.1 INTRODUCTION 5-1
5.2 REGISTERS 5-1
5.2.1 A/D CONTROL AND STATUS REGISTER 5-3
5.2.2 A/D DATA BUFFER REGISTER 5-6
5.2.3 CHANNEL-LIST PROGRAMMING REGISTER 5-6
5.2.4 DMA CONTROL AND STATUS REGISTER 5-8
5.2.5 DMA WORD COUNT REGISTER 5-11
5.2.6 DMA CURRENT ADDRESS REGISTER 5-11
5.3 MULTIPLEXER CHANNEL-LIST 5-11
5.4 ADDRESS POINTERS 5-12
5.5 EXTERNAL PORT 5-13
5.5.1 DT3369 DUAL-PORT MEMORY 5-14
5.5.2 SKY 320 SIGNAL PROCESSOR 5-15
5.6 DMA TRANSFERS 5-15
5.6.1 DMA TRANSFERS VIA Q-BUS 5-15
5.6.2 DMA TRANSFERS VIA EXTERNAL PORT 5-16

CHAPTER 6 OPERATING PRINCIPLES

6.1 INTRODUCTION 6-1
6.2 A/D CONVERSION TRIGGER 6-1
6.3 DATA TRANSFER 6-2
6.4 DATA CONVERSION 6-3

CHAPTER 7 CHANNEL-LIST PROGRAMMING

7.1 INTRODUCTION 7-1
7.2 PROGRAMMING PROTOCOLS 7-2

7.2.1 INITIALIZE POINTER LOGIC 7-2
7.2.2 WRITE START ADDRESS REGISTER/CURRENT ADDRESS POINTER 7-3
7.2.3 READ CURRENT ADDRESS POINTER 7-3
7.2.4 WRITING FINAL ADDRESS POINTER 7-4
7.2.5 READING THE FINAL ADDRESS POINTER 7-4
7.2.6 RESETTING THE POINTER 7-5
7.3 ENTERING CHANNEL INFORMATION INTO THE RAM 7-5
7.3.1 WRITING INTO THE CHANNEL-LIST RAM 7-6
7.3.2 READING THE CHANNEL-LIST RAM 7-7
7.3.3 MULTIPLEXER PRELOAD AND LOGIC ENABLE 7-8

CHAPTER 8 PROGRAMMING EXAMPLES

8.1 INTRODUCTION 8-1
8.2 FADEX1 - EXAMPLE PROGRAM 1 8-1
8.3 FADEX2 - EXAMPLE PROGRAM 2 8-1
8.4 FADEX3 - EXAMPLE PROGRAM 3 8-2
8.5 FADEX4 - EXAMPLE PROGRAM 4 8-2
8.6 FADEX5 - EXAMPLE PROGRAM 5 8-3

CHAPTER 9 A/D CALIBRATION

9.1 INTRODUCTION 9-1
9.2 GUIDELINES 9-2
9.2.1 PREREQUISITES 9-2
9.2.2 EQUIPMENT REQUIRED 9-3
9.2.3 PRECAUTIONS 9-3
9.2.4 SINGLE-ENDED INPUT OPERATION (SE) 9-3
9.2.5 DIFFERENTIAL INPUT OPERATION (DI) 9-4
9.3 CALIBRATION 9-5
9.3.1 UNIPOLAR RANGE CALIBRATION 9-8
9.3.1.1 Unipolar Zero 9-8
9.3.1.2 Unipolar Full Scale 9-9
9.3.1.3 PG Zero 9-9
9.3.2 BIPOLAR RANGE CALIBRATION 9-10
9.3.2.1 Minus Full Scale 9-10
9.3.2.2 Plus Full Scale 9-11
9.3.2.3 PG Zero 9-11
9.4 APPLICATIONS 9-12
9.5 GROUNDING 9-13
9.5.1 DIFFERENTIAL INPUTS 9-13
9.5.2 SINGLE-ENDED INPUTS 9-14
9.5.3 PSEUDO-DIFFERENTIAL INPUTS 9-14

APPENDIX A DT3362 SERIES JUMPER SUMMARY

A.1 INTRODUCTION A-1

APPENDIX B DATA CODING TABLES

B.1 INTRODUCTION B-1

APPENDIX C TROUBLESHOOTING

C.1 TROUBLESHOOTING C-1
C.1.1 SYMPTOMS AND SOLUTIONS C-2
C.1.1.1 No Board Response C-2
C.1.1.2 Digital Output Code In Plus Or Minus Full Scale Stops C-2
C.1.1.3 First Reading On A Channel Is Incorrect C-2
C.1.1.4 Zero Or Full Scale Will Not Calibrate C-2
C.2 RMA NUMBER C-2

APPENDIX D LISTING OF DT3362 SERIES DIAGNOSTICS

APPENDIX E ENGINEERING DRAWING

INDEX

FIGURES

3-1 BASE ADDRESS SELECTION OF 771300 (OCTAL). 3-4
3-2 VECTOR ADDRESS SELECTION 3-5

4-1 CONNECTION OF DT701-20 AND DT701-50 TO THE DT3362 SERIES BOARDS 4-7
4-2 CONNECTION OF DT709 SERIES AND DT6700 SERIES TO SE VERSIONS OF THE DT3362 SERIES BOARDS 4-8
4-3 CONNECTION OF DT756 SERIES WITH SE VERSIONS OF THE DT3362 SERIES EXCEPT DT3368-4SE AND DT3368-12SE . . . 4-8
4-4 CONNECTION OF SCREW TERMINAL PANELS TO THE A/D EXPANDER OF THE DT3362 SERIES 4-9
4-5 SINGLE-ENDED INPUT CONNECTION 4-12
4-6 PSEUDO-DIFFERENTIAL INPUT CONNECTION 4-14
4-7 DIFFERENTIAL INPUT CONNECTION 4-16

5-1 ADSCR BIT ASSIGNMENTS 5-3
5-2 CLPR BIT ASSIGNMENTS 5-7
5-3 DMACSR BIT ASSIGNMENTS 5-8
5-4 DATA PATH TAKEN BY CONVENTIONAL DMA TRANSFERS . . . 5-13
5-5 DATA PATH TAKEN WITH DUAL-PORTED MEMORY STRUCTURE . 5-14

9-1 SINGLE-ENDED CONNECTIONS (EXCEPT DT3368-4SE AND DT3368-12SE). 9-3
9-2 SINGLE-ENDED CONNECTIONS (DT3368-4SE AND DT3368-12SE) 9-4
9-3 DIFFERENTIAL CONNECTIONS 9-5

B-1 CALCULATING THE INPUT VOLTAGE TO AN A/D CONVERTER USING BINARY OR OFFSET BINARY CODING B-2
B-2 CALCULATING THE INPUT VOLTAGE TO AN A/D CONVERTER USING TWO'S COMPLEMENT CODING B-2

TABLES

1-1 DT3362 SERIES FEATURES COMPARISON 1-3

3-1 DT3362 SERIES INPUT MODES 3-6
3-2 INPUT MODE SELECTION 3-7
3-3 DT3362 SERIES INPUT RANGES 3-8
3-4 INPUT RANGE SELECTION 3-9
3-5 DT3362 SERIES OUTPUT CODES 3-11
3-6 SELECTION OF DATA CODING 3-12
3-7 FULL SCALE OUTPUT CODES IN VARIOUS NOTATIONS . . . 3-13
3-8 FULL SCALE RANGES FOR HIGH LEVEL INPUT BOARDS . . . 3-13
3-9 DMA WRAP MODE SELECTION 3-14
3-10 SELECTION OF DMA ADDRESSING 3-14
3-11 REFERENCE JUMPERS FOR REVISION F AND LATER BOARDS . 3-15
3-12 REFERENCE JUMPERS FOR ALL MODELS 3-16

4-1 J3 PIN ASSIGNMENTS ON ALL DT3362- PREFIXED MODELS . . 4-2
4-2 J2 EXPANDER PIN ASSIGNMENTS ON DT3362- PREFIXED MODELS 4-3
4-3 J3 PI ASSIGNMENTS ON DT3377 4-4
4-4 J3 PIN ASSIGNMENTS ON DT3368-4SE AND DT3368-14SE . . 4-4
4-5 J2 EXPANDER PIN ASSIGNMENTS ON DT3368-12SE 4-5
4-6 EXTERNAL PORT CONNECTOR J1 PIN ASSIGNMENTS 4-6
4-7 DT3362 SERIES INPUT TYPES 4-10

5-1 DT3362 SERIES REGISTER ASSIGNMENTS 5-2
5-2 CHANNEL-LIST CONTROL INSTRUCTION FIELD 5-4
5-3 CHANNEL-LIST OPERATION MODE FIELD 5-4
5-4 CHANNEL-LIST PAGE SELECT 5-6
5-5 MODE SELECTION USING CLPR 5-7
5-6 DMA MODE SELECTION 5-11

9-1 DT3362 SERIES A/D CONVERTER MODULES 9-2
9-2 1 LSB VOLTAGE LEVELS 9-6
9-3 12-BIT OFFSET BINARY AND TWO'S COMPLEMENT BINARY CODE FOR MINUS FULL SCALE PLUS 1 LSB 9-6
9-4 16-BIT TWO'S COMPLEMENT BINARY CODE FOR MINUS FULL SCALE PLUS 1 LSB 9-7
9-5 PLUS FULL SCALE MINUS 2 LSB 9-7
9-6 12-BIT BINARY CODES FOR PLUS FULL SCALE MINUS 2 LSB . 9-7
9-7 16-BIT TWO'S COMPLEMENT BINARY CODES FOR PLUS FULL SCALE MINUS 2 LSB 9-8
9-8 MINUS FULL SCALE PLUS ONE LSB VOLTAGE 9-10
9-9 12-BIT OFFSET BINARY AND TWO'S COMPLEMENT BINARY CODES FOR MINUS FULL SCALE PLUS 1 LSB 9-11
9-10 16-BIT TWO'S COMPLEMENT CODES FOR MINUS FULL SCALE PLUS 1 LSB 9-11
9-11 12-BIT OFFSET BINARY AND TWO'S COMPLEMENT BINARY OUTPUT CODES FOR 0 VOLTS 9-12
9-12 TYPICAL COUNT SPREAD DUE TO NOISE 9-12

A-1 DT3362-16SE/8DI AND DT3362-64SE/32DI JUMPERS A-2

A-2 DT3362-16SE/8DI-PGH AND DT3362-64SE/32DI-PGH JUMPERS A-3

A-3 DT3362-F-16SE AND DT3362-F-64SE JUMPERS A-4

A-4 DT3362-F-16DI AND DT3362-F-32DI JUMPERS A-5

A-5 DT3362-G-16SE AND DT3362-G-64SE JUMPERS A-6

A-6 DT3362-G-16DI AND DT3362-G-32DI JUMPERS A-7

A-7 DT3362-H-16SE AND DT3362-H-64SE JUMPERS A-8

A-8 DT3362-H-16DI AND DT3362-H-32DI JUMPERS A-9

A-9 DT3362-H-16SE-PGH AND DT3362-H-64SE-PGH JUMPERS . . A-10

A-10 DT3362-H-16DI-PGH AND DT3362-H-32DI-PGH JUMPERS . . A-11

A-11 DT3377 JUMPERS A-12

A-12 DT3368-4SE AND DT3368-12SE JUMPERS A-13

B-1 12-BIT A/D BIPOLAR CODING TABLE
(ALL MODELS EXCEPT DT3368-4SE AND DT3368-12SE). B-3

B-2 12-BIT A/D UNIPOLAR TABLE
(ALL MODELS EXCEPT DT3377) B-3

B-3 16-BIT A/D BIPOLAR CODING (DT3377) B-4

PREFACE

This hardware manual is written for users of the Data Translation DT3362 series data acquisition and conversion subsystems compatible with Digital Equipment Corporation's Q-bus-based computer systems. It describes the configuration of the board, shows system interconnections, discusses the board architecture, and provides you with information on how to program and operate the DT3362 series. The manual assumes that you have a general knowledge of electronic circuitry, are familiar with the Q-bus, understand data conversion, and possess enough proficiency in software to write your own application and diagnostic programs. The following list provides summaries of all sections in the manual.

CHAPTER 1 - INTRODUCTION

Chapter 1 provides an overview of all the models comprising the DT3362 series. The chapter briefly describes the major features of each model and gives a summary of compatible data acquisition accessories.

CHAPTER 2 - SPECIFICATIONS

Chapter 2 gives the complete electrical and physical specifications for all boards in the series. The chapter also lists compatible data acquisition accessories, connectors, and cable assemblies used to interface the DT3362 series board with signal sources.

CHAPTER 3 - UNPACKING AND CONFIGURATION

Chapter 3 describes the unpacking procedure and lists measures to take if product is damaged. The major portion of the chapter discusses the configuration of user-selectable parameters.

CHAPTER 4 - INTERCONNECTION

Chapter 4 discusses connections to the Q-bus and the user application. The chapter gives the bit assignments of all connectors and discusses input connection schemes. It also provides diagrams showing the connections of the DT3362 series boards to compatible screw terminal/signal conditioning panels.

CHAPTER 5 - ARCHITECTURE

Chapter 5 describes the board registers and the function of each bit in each register. The chapter also discusses the multiplexer channel-list, the external port, and DMA transfers.

CHAPTER 6 - OPERATING PRINCIPLES

Chapter 6 describes the various stages in the operation of the board--A/D conversion trigger, data transfer, and data conversion.

CHAPTER 7 - CHANNEL-LIST PROGRAMMING

Chapter 7 explains the use of the multiplexer channel-list feature and presents the various protocols and sequences needed to program the multiplexer channel-list and the pointer logic. The chapter also describes how to enter channel information into the RAM.

CHAPTER 8 - PROGRAMMING EXAMPLES

Chapter 8 presents five examples which describe and illustrate various programming aspects of the DT3362 series boards both for non-DMA applications as well as for DMA operations.

CHAPTER 9 - CALIBRATION

Chapter 9 describes calibration procedures for the A/D converters on the DT3362 series boards.

APPENDIX A - DT3362 SERIES JUMPER SUMMARY

Appendix A provides the jumper lists of all models in the DT3362 series boards.

APPENDIX B - DATA CODING TABLES

Appendix B provides unipolar and bipolar data coding tables for for key points in the input ranges of all models in the DT3362 series boards.

APPENDIX C - TROUBLESHOOTING

Appendix C lists the steps to be taken in troubleshooting the DT3362 series board and in seeking factory service.

APPENDIX D - LISTING OF DT3362 SERIES DIAGNOSTICS

Appendix D contains the listing of the diagnostics.

APPENDIX E - ENGINEERING DRAWING

Appendix E contains the board assembly drawing of the DT3362 series.

CHAPTER 1

INTRODUCTION

The DT3362 series of analog to digital (A/D) converter boards are fast, Q-bus compatible data acquisition subsystems intended for use in a wide range of applications in industrial process control as well as laboratory research. The series offers such advanced features as 22-bit addressing, data transfers via DMA, and an external port to enhance throughput.

The series includes basic boards and expanded boards. The maximum number of input channels on the basic boards is 16 single-ended or 8 differential. Expanded boards are basic boards equipped with expander modules to provide additional input channels. Up to 64 single-ended or 32 differential input channels can be accommodated on expanded boards. The series uses 12-bit, as well as 16-bit A/D converters, and features boards with simultaneous sample and hold circuits.

The DT3362 series boards are designed for both unipolar and bipolar inputs. The unipolar input range is 0 to +10V for models not equipped with programmable gain (PGH) and 0 to +1.25V to 0 to +10V for models with programmable gain. The corresponding bipolar ranges are ±10V and ±1.25V to ±10V, respectively. The programmable gain is software selectable. The high resolution (16-bit) model in the series operates only in a bipolar input range.

Input modes on the DT3362 series boards can be single-ended or differential. Four models offer selectable input modes: DT3362-16SE/8DI, DT3362-64SE/32DI, DT3362-16SE/8DI-PGH, and DT3362-64SE/32DI-PGH. The other models are configured for either differential or single-ended inputs.

A conversion trigger initiates a single conversion, a scan of conversions in a channel-list, or a specified number of conversions, all under software control. Conversion triggers are issued either via software or by an external pulse.

The sophisticated architecture of the DT3362 series permits data transfers to memory over a dedicated bus which is completely independent of the system Q-bus. This feature is used in combination with the DT3369 dual port memory board to further enhance subsystem throughput without adding to the latency of the Q-bus, thus freeing the system bus to service other peripherals.

The DT3362 series boards require no external power to operate. On-board dc to dc converters generate +15V and -15V power required by the A/D converter modules. The dc to dc converters and logic circuits obtain power directly from the +5V supply line on the Q-bus backplane.

The DT3362 series board occupies 16 consecutive byte locations in the Q-bus I/O address space. The base address can be assigned any value within the 760000-777760 (octal) range in increments of 20 (octal).

1.1 MODEL DESCRIPTIONS

The DT3362 series consists of 23 models: 12 basic models and 11 expanded. Table 1-1 lists all the models and provides a summary of their performance characteristics.

TABLE 1-1: DT3362 SERIES FEATURES COMPARISON

MODEL	CONV. RATE(1)	RESOLUTION	ANALOG INPUTS	SOFTWARE GAIN
DT3362-16SE/8DI	50kHz	12 bits	16SE/8DI	N/A
DT3362-32DI/64SE (1)	50kHz	12 bits	64SE/32DI	N/A
DT3362-16SE/8DI-PGH	50kHz	12 bits	16SE/8DI	1,2,4,8
DT3362-64SE/32DI-PGH (2)	50kHz	12 bits	64SE/32DI	1,2,4,8
DT3362-F-16SE	125kHz	12 bits	16SE	1,2,4,8
DT3362-F-64SE (2)	125kHz	12 bits	64SE	1,2,4,8
DT3362-F-8DI	125kHz	12 bits	8DI	1,2,4,8
DT3362-F-32DI (2)	125kHz	12 bits	32DI	1,2,4,8
DT3362-G-16SE	250kHz	12 bits	16SE	1,2,4,8
DT3362-G-64SE (2)	250kHz	12 bits	64SE	1,2,4,8
DT3362-G-8DI	250kHz	12 bits	8DI	1,2,4,8
DT3362-G-32DI (2)	250kHz	12 bits	32DI	1,2,4,8
DT3362-H-16SE	250kHz	12 bits	16SE	N/A
DT3362-H-64SE (2)	250kHz	12 bits	64SE	N/A
DT3362-H-8DI	250kHz	12 bits	8DI	N/A
DT3362-H-32DI (2)	250kHz	12 bits	32DI	N/A
DT3362-H-16SE-PGH	200kHz	12 bits	16SE	1,2,4,8
DT3362-H-64SE-PGH (2)	200kHz	12 bits	64SE	1,2,4,8
DT3362-H-8DI-PGH	200kHz	12 bits	8DI	1,2,4,8
DT3362-H-32DI-PGH (2)	200kHz	12 bits	32DI	1,2,4,8
DT3377	100kHz	16 bits	4DI	N/A
DT3368-4SE (3)	100kHz	12 bits	4SE	N/A
DT3368-12SE (2,3)	100kHz	12 bits	12SE	N/A

NOTES:

1. "Conversion Rate" is the maximum number of A/D conversions per second of which the board is capable as shipped from the factory. This does not take into account software latencies and the time required to read data from the board.
2. Uses an expander module.
3. Uses a simultaneous sample and hold module.

A brief description of all the models in the DT3362 series follows.

DT3362-16SE/8DI

High level analog input system with 12-bit resolution and maximum 50kHz throughput. Jumper-selectable for 16 single-ended or 8 differential input channels. Accepts unipolar or bipolar inputs. Factory-configured for differential inputs and bipolar input range.

DT3362-64SE/32DI
Same as DT3362-16SE/8DI, but expanded to 64SE/32DI input channels.

DT3362-16SE/8DI-PGH
High level analog input system with 12-bit resolution and maximum 50kHz throughput. Jumper selectable for 16 single-ended or 8 differential input channels. Accepts unipolar or bipolar inputs. Software programmable for gains of 1, 2, 4, or 8. Factory-configured for differential inputs and bipolar input range.

DT3362-64SE/32DI-PGH
Same as DT3362-16SE/8DI-PGH, but expanded to 64SE/32DI input channels.

DT3362-F-16SE
Fast, high level analog input system with 12-bit resolution and maximum 125kHz throughput. Provides 16 single-ended input channels. Accepts unipolar or bipolar inputs. Software programmable for gains of 1, 2, 4, or 8.

DT3362-F-64SE
Same as DT3362-F-16SE, but expanded to 64SE input channels.

DT3362-F-8DI
Fast, high level analog input system with 12-bit resolution and maximum 125kHz throughput. Provides 8 differential input channels. Accepts unipolar or bipolar inputs. Software programmable for gains of 1, 2, 4, or 8.

DT3362-F-32DI
Same as DT3362-F-8DI, but expanded to 32DI input channels.

DT3362-G-16SE
Fast, high level analog input system with 12-bit resolution and maximum 250kHz throughput. Provides 16 single-ended input channels. Accepts unipolar or bipolar inputs. Software programmable for gains of 1, 2, 4, or 8.

DT3362-G-64SE
Same as DT3362-G-16SE, but expanded to 64DI input channels.

DT3362-G-8DI
Fast, high level analog input system with 12-bit resolution and maximum 250kHz throughput. Provides 8 differential input channels. Accepts unipolar or bipolar inputs. Software programmable for gains of 1, 2, 4, or 8.

DT3362-G-32DI
Same as DT3362-G-8DI, but expanded to 32DI input channels.

DT3362-H-16SE
Fast, high level analog input system with 12-bit resolution and maximum 250kHz throughput. Provides 16 single-ended input channels. Accepts unipolar or bipolar inputs.

DT3362-H-64SE
Same as DT3362-H-16SE, but expanded to 64SE input channels.

DT3362-H-8DI
Fast, high level analog input system with 12-bit resolution and maximum 250kHz throughput. Provides 8 differential input channels. Accepts unipolar or bipolar inputs.

DT3362-H-32DI
Same as DT3362-H-8DI-PGH, but expanded to 32DI input channels.

DT3362-H-16SE-PGH
Fast, high level analog input system with 12-bit resolution and maximum 200kHz throughput. Provides 16 single-ended input channels. Accepts unipolar or bipolar inputs. Software programmable for gains of 1, 2, 4, or 8.

DT3362-H-64SE-PGH
Same as DT3362-H-16SE-PGH, but expanded to 64SE input channels.

DT3362-H-8DI-PGH
Fast, high level analog input system with 12-bit resolution and maximum 200kHz throughput. Provides 8 differential input channels. Accepts unipolar or bipolar inputs. Software programmable for gains of 1, 2, 4, or 8.

DT3362-H-32DI
Same as DT3362-H-8DI-PGH, but expanded to 32DI input channels.

DT3368-4SE
High level analog input system featuring a simultaneous sample and hold converter module. Provides 4 single-ended input channels. Offers 12-bit resolution and 100kHz throughput. Accepts unipolar or bipolar inputs. Operates at a gain of 1.

DT3368-12SE
Same as DT3368-4SE, but expanded to 12SE input channels.

DT3377
Fast, high level analog input system with 16-bit resolution and 100kHz throughput. Provides 4 differential input channels. Accepts bipolar inputs only. Operates at a gain of 1.

1.2 COMPATIBLE SCREW TERMINAL PANELS

The DT3362 series boards are compatible with a number of data acquisition accessories--screw terminal panels--available from Data Translation. The data acquisition accessories belong in two groups: those that simply interface the A/D board to the signal source; and those that condition the input signals in addition to providing a connection interface. The following is a brief description of screw terminal panels that can be used with the various models in the DT3362 series.

1.2.1 DT701 SERIES

The DT701 series of screw terminal panels are compatible with all models of the DT3362 series. Two versions, the DT701-20 and DT701-50, can be used. The DT701-20 is used with the J3 connector of the basic or expanded versions of the DT3362 series, while the DT701-50 is used with the expanded versions connecting to J2. These panels provide standard matrix patterns for user-installation of conditioning circuits on each input channel. The matrix pattern accommodates current loop resistors, noise filters, voltage dividers, open thermocouple detection circuit, or any circuit required which mates with the standard pattern.

1.2.2 DT6700 SERIES

The DT6700 series of isolated signal conditioning modules are compatible with all single-ended models of the DT3362 series. They are suited particularly for industrial application where noise and common mode voltage could cause serious problems. The modules are mounted on the DT750 backplane which connects to the J3 connector of the basic or expanded versions of the DT3362 series. The J2 connection of the expanded board accommodates three DT750 backplanes. The conversion of the 50-pin J2 connector to three 20-pin connectors of the screw terminal panels is made via the EP164 interconnection panel.

1.2.3 DT709 SERIES

The DT709-S and DT709-Y of the DT709 series of screw terminal/signal conditioning panels are compatible with all single-ended models of the DT3362 series. The DT709-S provides only connection points to the measurement, while the DT709-Y also offers common mode rejection and other signal conditioning functions. As a result, the DT709-Y effectively converts the single-ended inputs of the DT3362 series into differential inputs. The basic versions of the DT3362 series accommodate one DT709 series screw terminal panel via the J3 connector. The expanded versions accommodate one panel via the J3 connector and three panels via the J2 connector using the EP164 interconnection panel. The DT709-Y requires an external power source capable of supplying ±11V to ±15V at ±30mA minimum.

1.2.4 DT756 SERIES

The DT756-Y and DT756-D of the DT756 series of screw terminal/signal conditioning panels are compatible with all single-ended models of the DT3362 series except the DT3368-4SE and DT3368-12SE which feature the simultaneous sample and hold module. Both models provide signal conditioning functions, including common mode rejection which converts the single-ended channels of the A/D board into differential channels. The basic versions of the DT3362 series accommodate one DT756 series screw terminal panel via the J3 connector. The expanded versions accommodate one panel via the J3 connector and three panels via the J2 connector using the EP164 interconnection panel as shown in Figure 1-1. The DT756-Y requires an external power source capable of supplying ±11V to ±15V at ±30mA minimum. The DT756-D has an on-board dc to dc converter and requires only a +5V input at 250mA.

CHAPTER 2

SPECIFICATIONS

2.1 INTRODUCTION

This chapter lists the specifications of the DT3362 series boards and related accessories. Specifications are provided for the following items:

- Analog Inputs
- Accuracy
- Dynamic Performance
- Thermal Characteristics
- On-board Clock
- External Trigger
- Interface Characteristics
- Power Requirements
- Physical/Environmental
- Connectors and Cables
- Compatible Screw Terminal Panels

Unless noted otherwise, the specifications are typical at +25°C and rated voltage.

2.2 DT3362-16SE/8DI AND DT3362-64SE/32DI

ANALOG INPUTS

Number of Analog Inputs	
DT3362-16SE/8DI	16SE or 8DI, user-selectable
DT3362-64SE/32DI	64SE or 32DI, user-selectable
Input Ranges	0 to +10V (unipolar); ±10V (bipolar)

Output Data Codes	Straight binary (unipolar);
	Offset binary (bipolar);
	Two's complement (bipolar);
	jumper-selectable.
	Sign extension available with bipolar ranges

Input Impedance	"Off" channel: 100MΩ in parallel with 10pF; "On" channel: 100Ω in parallel with 100pF
Bias Current	±20nA
Common Mode Input Voltage, Maximum	±11V
Common Mode Rejection Ratio (CMRR), Gain = 1	80dB at 60Hz, 1kΩ unbalanced
Maximum Input Voltage Without Damage, Power On	±35V
Maximum Input Voltage Without Damage, Power Off	±20V
Amplifier Input Noise	0.2 LSB rms NOTE: Where gain is greater than 1, input noise is multiplied by gain.
Channel-to-channel Input Voltage Error	±5μV

ACCURACY

Resolution	12 bits
Nonlinearity	Less than ±1/2 LSB
Differential Nonlinearity	Less than ±1/2 LSB
Inherent Quantizing Error	±1/2 LSB
System Accuracy	To within ±.03% FSR
Channel Cross Talk	-80dB at 1kHz
Gain Error	Adjustable to 0
Zero Error	Adjustable to 0

DYNAMIC PERFORMANCE

Channel Acquisition Time To within 1/2 LSB	15μs
A/D Conversion Time	10μs
Max. A/D Converter Throughput	50kHz (Throughput is defined as the maximum frequency of A/D conversions achievable by the board. The figure given above does not allow for software overhead necessary to retrieve the converted data.)
Sample And Hold Aperture Uncertainty	10ns
Sample And Hold Aperture Delay	50ns
Sample And Hold Feedthrough Attenuation	80dB at 1kHz
Sample And Hold Droop Rate	0.1mV/ms

THERMAL CHARACTERISTICS

A/D Zero Drift	±10μV/°C (unipolar); ±10ppm of FSR/°C (bipolar)
Amplifier Zero Drift	[±20μV/°C] + [(±3μV/°C)xGain]
Gain Drift	±30ppm of FSR/°C
Differential Linearity Drift	±3ppm of FSR/°C
Monotonicity	Monotonic, 0 to +50°C (32 to 122°F)

2.3 DT3362-16SE/8DI-PGH AND DT3362-64SE/32DI-PGH

ANALOG INPUTS

Number of Analog Inputs	
DT3362-8DI/16SE-PGH	8DI or 16SE, user-selectable
DT3362-32DI/64SE-PGH	32DI or 64SE, user-selectable
Input Ranges	0 to 1.25V to 0 to +10V (unipolar); ±1.25V to ±10V (bipolar)

Output Data Codes	Straight binary (unipolar);
	Offset binary (bipolar);
	Two's complement (bipolar); jumper selectable. Sign extension is available with bipolar ranges

Software Programmable Gain	1, 2, 4, 8
Input Impedance	"Off" channel: 100MΩ in parallel with 10pF; "On" channel: 100MΩ in parallel with 100pF
Bias Current	±20nA
Common Mode Input Voltage, Maximum	±11V
Common Mode Rejection Ratio (CMRR), Gain = 1	80dB at 60Hz, 1kΩ unbalanced
Maximum Input Voltage Without Damage, Power On	±35V
Maximum Input Voltage Without Damage, Power Off	±20V
Amplifier Input Noise	0.2 LSB rms NOTE: Where gain is greater than 1, input noise is multiplied by gain.
Channel-to-channel Input Voltage Error	±5μV

ACCURACY

Resolution	12 bits
Nonlinearity	Less than ±1/2 LSB
Differential Nonlinearity	Less than ±1/2 LSB
Inherent Quantizing Error	±1/2 LSB
System Accuracy	
Gain = 1	To within ±.03% FSR
Gain = 8	To within ±.05% FSR
Channel Cross Talk	-80dB at 1kHz
Gain Error	Adjustable to 0
Zero Error	Adjustable to 0

DYNAMIC PERFORMANCE

Channel Acquisition Time To within 1/2 LSB	15μs
A/D Conversion Time	10μs
Max. A/D Converter Throughput	50kHz (Throughput is defined as the maximum frequency of A/D conversions achievable by the board. The figure given above does not allow for software overhead necessary to retrieve the converted data.)
Sample And Hold Aperture Uncertainty	10ns
Sample And Hold Aperture Delay	50ns
Sample And Hold Feedthrough Attenuation	80dB at 1kHz
Sample And Hold Droop Rate	0.1mV/ms

THERMAL CHARACTERISTICS

A/D Zero Drift	±10μV/°C (unipolar) ±10ppm of FSR/°C (bipolar)
Amplifier Zero Drift	[±20μV/°C] + [(±3μV/°C)xGain]
Gain Drift	±30ppm of FSR/°C
Differential Linearity Drift	±3ppm of FSR/°C
Monotonicity	Monotonic, 0 to +50°C (32 to 122°C)

2.4 DT3362-F-16SE AND DT3362-F-64SE

ANALOG INPUTS

Number of Analog Inputs	
DT3362-F-16SE	16SE
DT3362-F-64SE	64SE
Input Ranges	0 to 1.25V to 0 to +10V (unipolar); ±1.25V to ±10V (bipolar)
Output Data Codes	Straight binary (unipolar) Offset binary (bipolar); Two's complement (bipolar); jumper-selectable. Sign extension is available with bipolar ranges
Software Programmable Gain	1, 2, 4, 8
Input Impedance	"Off" channel: 100MΩ in parallel with 50pF; "On" channel: 100MΩ in parallel with 100pF
Bias Current	10nA
Common Mode Input Voltage, Maximum	±10.5V
Common Mode Rejection Ratio (CMRR), Gain = 1	80dB at 60Hz, 1kΩ unbalanced
Maximum Input Voltage Without Damage, Power On	±27V
Maximum Input Voltage Without Damage, Power Off	±12V
Amplifier Input Noise	0.2 LSB rms NOTE: Where gain is greater than 1, input noise is multiplied by gain.
Channel-to-channel Input Voltage Error	1/2 LSB

ACCURACY

Resolution	12 bits
Nonlinearity	Less than ±1/2 LSB
Differential Nonlinearity	Less than 0.012% of FSR
Inherent Quantizing Error	±1/2 LSB
System Accuracy	
Gain = 1	To within ±.03% FSR
Gain = 2	To within ±.04% FSR
Gain = 4	To within ±.05% FSR
Gain = 8	To within ±.07% FSR
Channel Cross Talk	-100dB at 1kHz
Gain Error	Adjustable to 0
Zero Error	Adjustable to 0
Offset Error	Adjustable to 0
Noise	0.2 LSB rms

DYNAMIC PERFORMANCE

Channel Acquisition Time To within 1/2 LSB	4μs
A/D Conversion Time	4μs
Max. A/D Converter Throughput	125kHz (Throughput is defined as the maximum frequency of A/D conversions achievable by the board. The figure given above does not allow for software overhead necessary to retrieve the converted data.)
Sample And Hold Acquisition Time	2.5μs
Sample And Hold Aperture Uncertainty	0.5ns
Sample And Hold Aperture Delay	100ns
Sample And Hold Feedthrough Attenuation	80dB at 1kHz
Sample And Hold Droop Rate	50μV/μs

THERMAL CHARACTERISTICS

A/D Zero Drift	±50μV/°C (unipolar) ±10ppm of FSR/°C (bipolar)
Amplifier Zero Drift	50μV/°C
Gain Drift	±30ppm of FSR/°C
Differential Linearity Drift	±3ppm of FSR/°C
Monotonicity	Monotonic, 0 to +70°C (32 to 158°C)

2.5 DT3362-F-8DI AND DT3362-F-32DI

ANALOG INPUTS

Number of Analog Inputs	
DT3362-F-8DI	8DI
DT3362-F-32DI	32DI
Input Ranges	0 to 1.25V to 0 to +10V (unipolar); ±1.25V to ±10V (bipolar)
Output Data Codes	Straight binary (unipolar) Offset binary (bipolar); Two's complement (bipolar); jumper selectable. Sign extension is available with bipolar ranges
Software Programmable Gain	1, 2, 4, 8
Input Impedance	"Off" channel: 100MΩ in parallel with 50pF; "On" channel: 100MΩ in parallel with 100pF
Bias Current	10nA
Common Mode Input Voltage, Maximum	±10.5V
Common Mode Rejection Ratio (CMRR), Gain = 1	80dB at 60Hz, 1kΩ unbalanced
Maximum Input Voltage Without Damage, Power On	±27V
Maximum Input Voltage Without Damage, Power Off	±12V
Amplifier Input Noise	0.2 LSB rms NOTE: Where gain is greater than 1, input noise is multiplied by gain.
Channel-to-channel Input Voltage Error	1/2 LSB

ACCURACY

Resolution	12 bits
Nonlinearity	Less than ±1/2 LSB
Differential Nonlinearity	Less than 0.012% of FSR
Inherent Quantizing Error	±1/2 LSB
System Accuracy	
Gain = 1	To within ±.03% FSR
Gain = 2	To within ±.04% FSR
Gain = 4	To within ±.05% FSR
Gain = 8	To within ±.07% FSR
Channel Cross Talk	-100dB at 1kHz
Gain Error	Adjustable to 0
Zero Error	Adjustable to 0
Offset Error	Adjustable to 0
Noise	0.2 LSB rms

DYNAMIC PERFORMANCE

Channel Acquisition Time To within 1/2 LSB	4µs
A/D Conversion Time	4µs
Max. A/D Converter Throughput	125kHz (Throughput is defined as the maximum frequency of A/D conversions achievable by the board. The figure given above does not allow for software overhead necessary to retrieve the converted data.)
Sample And Hold Acquisition Time	2.5µs
Sample And Hold Aperture Uncertainty	0.5ns
Sample And Hold Aperture Delay	100ns
Sample And Hold Feedthrough Attenuation	80dB at 1kHz
Sample And Hold Droop Rate	50µV/µs

THERMAL CHARACTERISTICS

A/D Zero Drift	±50µV/°C (unipolar) ±10ppm of FSR/°C (bipolar)
Amplifier Zero Drift	50µV/°C
Gain Drift	±30ppm of FSR/°C
Differential Linearity Drift	±3ppm of FSR/°C
Monotonicity	Monotonic, 0 to +70°C (32 to 158°F)

2.6 DT3362-G-16SE AND DT3362-G-64SE

ANALOG INPUTS

Number of Analog Inputs	
DT3362-G-16SE	16SE
DT3362-G-64SE	64SE
Input Ranges	0 to 1.25V to 0 to +10V (unipolar); ±1.25V to ±10V (bipolar)

Output Data Codes	Straight binary (unipolar);
	Offset binary (bipolar);
	Two's complement (bipolar); jumper selectable. Sign extension is available with bipolar ranges

Software Programmable Gain	1, 2, 4, 8
Input Impedance	"Off" channel: 100MΩ in parallel with 10pF; "On" channel: 100MΩ in parallel with 50pF
Bias Current	±20nA
Common Mode Input Voltage, Maximum	±11V
Common Mode Rejection Ratio (CMRR), Gain = 1	80dB at 60Hz, 1kΩ unbalanced
Maximum Input Voltage Without Damage, Power On	±16V
Maximum Input Voltage Without Damage, Power Off	±1V
Amplifier Input Noise	100μV rms
Channel-to-channel Input Voltage Error	±100μV

ACCURACY

Resolution	12 bits
Nonlinearity	Less than ±1/2 LSB
Differential Nonlinearity	Less than ±1/2 LSB
Inherent Quantizing Error	±1/2 LSB
System Accuracy	
Gain = 1	To within ±.03% FSR
Gain = 8	To within ±.05% FSR
Channel Cross Talk	-80dB at 1kHz
Gain Error	Adjustable to 0
Zero Error	Adjustable to 0
Noise	0.2 LSB rms

DYNAMIC PERFORMANCE

Channel Acquisition Time	
To within 1/2 LSB	2.5μs
A/D Conversion Time	2.5μs
Max. A/D Converter Throughput	250kHz (Throughput is defined as the maximum frequency of A/D conversions achievable by the board. The figure given above does not allow for software overhead necessary to retrieve the converted data.)
Sample And Hold Aperture Uncertainty	Less than 2ns
Sample And Hold Aperture Delay	40ns
Sample _Hold Feedthrough Attenuation	80dB at 1kHz
Sample And Hold Droop Rate	20μV/μs

SPECIFICATIONS
DT3362-G-16SE AND DT3362-G-64SE

THERMAL CHARACTERISTICS

A/D Zero Drift	±50μV/°C (unipolar); ±25ppm of FSR/°C (bipolar)
Amplifier Zero Drift	[±20μV/°C] + [(±10μV/°C)xGain]
Gain Drift	±40ppm of FSR/°C
Differential Linearity Drift	±3ppm of FSR/°C
Monotonicity	Monotonic, 0 to +50°C (32 to 122°F)

2.7 DT3362-G-8DI AND DT3362-G-32DI

ANALOG INPUTS

Number of Analog Inputs	
DT3362-G-8DI	8DI
DT3362-G-32DI	32DI
Input Ranges	0 to 1.25V to 0 to +10V (unipolar); ±1.25V to ±10V (bipolar)
Output Data Codes	Straight binary (unipolar) Offset binary (bipolar); Two's complement (bipolar); jumper selectable. Sign extension is available with bipolar ranges
Software Programmable Gain	1, 2, 4, 8
Input Impedance	"Off" channel: 100MΩ in parallel with 10pF; "On" channel: 100Ω in parallel with 50pF
Bias Current	±20nA
Common Mode Input Voltage, Maximum	±11V
Common Mode Rejection Ratio (CMRR), Gain = 1	80dB at 60Hz, 1kΩ unbalanced
Maximum Input Voltage Without Damage, Power On	±16V
Maximum Input Voltage Without Damage, Power Off	±1V
Amplifier Input Noise	100μV rms
Channel-to-channel Input Voltage Error	±100μV

ACCURACY

Resolution	12 bits
Nonlinearity	Less than ±1/2 LSB
Differential Nonlinearity	Less than ±1/2 LSB
Inherent Quantizing Error	±1/2 LSB
System Accuracy	
Gain = 1	To within ±.03% FSR
Gain = 8	To within ±.05% FSR
Channel Cross Talk	-80dB at 1kHz
Gain Error	Adjustable to 0
Zero Error	Adjustable to 0
Noise	0.2 LSB rms

DYNAMIC PERFORMANCE

Channel Acquisition Time To within 1/2 LSB	2.5μs
A/D Conversion Time	2.5μs
Max. A/D Converter Throughput	250kHz (Throughput is defined as the maximum frequency of A/D conversions achievable by the board. The figure given above does not allow for software overhead necessary to retrieve the converted data.)
Sample And Hold Aperture Uncertainty	Less than 2ns
Sample And Hold Aperture Delay	40ns
Sample _Hold Feedthrough Attenuation	80dB at 1kHz
Sample And Hold Droop Rate	20μV/μs

THERMAL CHARACTERISTICS

A/D Zero Drift	±50μV/°C (unipolar); ±25ppm of FSR/°C (bipolar)
Amplifier Zero Drift	[±20μV/°C] + [(±10μV/°C)xGain]
Gain Drift	±40ppm of FSR/μC
Differential Linearity Drift	±3ppm of FSR/μC
Monotonicity	Monotonic, 0 to +50°C (32 to 122°F)

2.8 DT3362-H-16SE AND DT3362-H-64SE

ANALOG INPUTS

Number of Analog Inputs	
DT3362-H-16SE	16SE
DT3362-H-64SE	64SE
Input Ranges	0 to +10V (unipolar); ±10V (bipolar)
Output Data Codes	Straight binary (unipolar); Offset binary (bipolar); Two’s complement (bipolar); jumper selectable. Sign extension is available with bipolar ranges
Input Impedance	"Off" channel: 100MΩ in parallel with 10pF; "On" channel: 100MΩ in parallel with 50pF
Bias Current	±20nA
Common Mode Input Voltage, Maximum	±11V
Common Mode Rejection Ratio (CMRR)	80dB at 60Hz, 1kΩ unbalanced
Maximum Input Voltage Without Damage, Power On	±16V
Maximum Input Voltage Without Damage, Power Off	±1V
Amplifier Input Noise	100μV rms
Channel-to-channel Input Voltage Error	±100μV

ACCURACY

Resolution	12 bits
Nonlinearity	Less than ±1/2 LSB
Differential Nonlinearity	Less than ±1/2 LSB
Inherent Quantizing Error	±1/2 LSB
System Accuracy	To within ±.03% FSR
Channel Cross Talk	-80dB at 1kHz
Gain Error	Adjustable to 0
Zero Error	Adjustable to 0
Noise	0.2 LSB rms

DYNAMIC PERFORMANCE

Channel Acquisition Time To within 1/2 LSB	2.5μs
A/D Conversion Time	2.5μs
Max. A/D Converter Throughput	250kHz (Throughput is defined as the maximum frequency of A/D conversions achievable by the board. The figure given above does not allow for software overhead necessary to retrieve the converted data.)
Sample And Hold Aperture Uncertainty	Less than 2ns
Sample And Hold Aperture Delay	40ns
Sample _Hold Feedthrough Attenuation	80dB at 1kHz
Sample And Hold Droop Rate	20μV/μs

THERMAL CHARACTERISTICS

A/D Zero Drift	±50µV/°C (unipolar); ±25ppm of FSR/°C (bipolar)
Amplifier Zero Drift	[±20µV/°C] + [(±10µV/°C)xGain]
Gain Drift	±40ppm of FSR/°C
Differential Linearity Drift	±3ppm of FSR/°C
Monotonicity	Monotonic, 0 to +50°C (32 to 122°F)

2.9 DT3362-H-8DI AND DT3362-H-32DI

ANALOG INPUTS

Number of Analog Inputs	
DT3362-H-8DI	8DI
DT3362-H-32DI	32DI
Input Ranges	0 to +10V (unipolar); ±10V (bipolar)
Output Data Codes	Straight binary (unipolar) Offset binary (bipolar); Two's complement (bipolar); jumper selectable. Sign extension is available with bipolar ranges
Input Impedance	"Off" channel: 100MΩ in parallel with 10pF; "On" channel: 100Ω in parallel with 50pF
Bias Current	±20nA
Common Mode Input Voltage, Maximum	±11V
Common Mode Rejection Ratio (CMRR)	80dB at 60Hz, 1kΩ unbalanced
Maximum Input Voltage Without Damage, Power On	±16V
Maximum Input Voltage Without Damage, Power Off	±1V
Amplifier Input Noise	100μV rms
Channel-to-channel Input Voltage Error	±100μV

ACCURACY

Resolution	12 bits
Nonlinearity	Less than ±1/2 LSB
Differential Nonlinearity	Less than ±1/2 LSB
Inherent Quantizing Error	±1/2 LSB
System Accuracy	To within ±.03% FSR

Channel Cross Talk	-80dB at 1kHz
Gain Error	Adjustable to 0
Zero Error	Adjustable to 0
Noise	0.2 LSB rms

DYNAMIC PERFORMANCE

Channel Acquisition Time To within 1/2 LSB	2.5μs
A/D Conversion Time	2.5μs
Max. A/D Converter Throughput	250kHz (Throughput is defined as the maximum frequency of A/D conversions achievable by the board. The figure given above does not allow for software overhead necessary to retrieve the converted data.)
Sample And Hold Aperture Uncertainty	Less than 2ns
Sample And Hold Aperture Delay	40ns
Sample _Hold Feedthrough Attenuation	80dB at 1kHz
Sample And Hold Droop Rate	20μV/μs

THERMAL CHARACTERISTICS

A/D Zero Drift	±50μV/°C (unipolar); ±25ppm of FSR/°C (bipolar)
Amplifier Zero Drift	[±20μV/°C] + [(±10μV/°C)xGain]
Gain Drift	±40ppm of FSR/μC
Differential Linearity Drift	±3ppm of FSR/μC
Monotonicity	Monotonic, 0 to +50°C (32 to 122°F)

2.10 DT3362-H-16SE-PGH AND DT3362-H-64SE-PGH

ANALOG INPUTS

Number of Analog Inputs	
DT3362-H-16SE-PGH	16SE
DT3362-H-64SE-PGH	64SE
Input Ranges	0 to 1.25V to 0 to +10V (unipolar); ±1.25V to ±10V (bipolar)
Output Data Codes	Straight binary (unipolar); Offset binary (bipolar); Two's complement (bipolar); jumper selectable. Sign extension is available with bipolar ranges
Software Programmable Gain	1, 2, 4, 8
Input Impedance	"Off" channel: 100MΩ in parallel with 10pF; "On" channel: 100MΩ in parallel with 50pF
Bias Current	±20nA
Common Mode Input Voltage, Maximum	±11V
Common Mode Rejection Ratio (CMRR), Gain = 1	80dB at 60Hz, 1kΩ unbalanced
Maximum Input Voltage Without Damage, Power On	±16V
Maximum Input Voltage Without Damage, Power Off	±1V
Amplifier Input Noise	100µV rms
Channel-to-channel Input Voltage Error	±100µV

ACCURACY

Resolution	12 bits
Nonlinearity	Less than ±1/2 LSB
Differential Nonlinearity	Less than ±1/2 LSB
Inherent Quantizing Error	±1/2 LSB
System Accuracy	
Gain = 1	To within ±.03% FSR
Gain = 8	To within ±.05% FSR
Channel Cross Talk	-80dB at 1kHz
Gain Error	Adjustable to 0
Zero Error	Adjustable to 0
Noise	0.2 LSB rms

DYNAMIC PERFORMANCE

Channel Acquisition Time To within 1/2 LSB	2.5μs
A/D Conversion Time	2.5μs
Max. A/D Converter Throughput	200kHz (Throughput is defined as the maximum frequency of A/D conversions achievable by the board. The figure given above does not allow for software overhead necessary to retrieve the converted data.)
Sample And Hold Aperture Uncertainty	Less than 2ns
Sample And Hold Aperture Delay	40ns
Sample _Hold Feedthrough Attenuation	80dB at 1kHz
Sample And Hold Droop Rate	20μV/μs

THERMAL CHARACTERISTICS

A/D Zero Drift	±50µV/°C (unipolar); ±25ppm of FSR/°C (bipolar)
Amplifier Zero Drift	[±20µV/°C] + [(±10µV/°C)xGain]
Gain Drift	±40ppm of FSR/°C
Differential Linearity Drift	±3ppm of FSR/°C
Monotonicity	Monotonic, 0 to +50°C (32 to 122°F)

2.11 DT3362-H-8DI-PGH AND DT3362-H-32DI-PGH

ANALOG INPUTS

Number of Analog Inputs	
DT3362-H-8DI-PGH	8DI
DT3362-H-32DI-PGH	32DI
Input Ranges	0 to 1.25V to 0 to +10V (unipolar); ±1.25V to ±10V (bipolar)
Output Data Codes	Straight binary (unipolar) Offset binary (bipolar); Two's complement (bipolar); jumper selectable. Sign extension is available with bipolar ranges
Software Programmable Gain	1, 2, 4, 8
Input Impedance	"Off" channel: 100MΩ in parallel with 10pF; "On" channel: 100Ω in parallel with 50pF
Bias Current	±20nA
Common Mode Input Voltage, Maximum	±11V
Common Mode Rejection Ratio (CMRR), Gain = 1	80dB at 60Hz, 1kΩ unbalanced
Maximum Input Voltage Without Damage, Power On	±16V
Maximum Input Voltage Without Damage, Power Off	±1V
Amplifier Input Noise	100μV rms
Channel-to-channel Input Voltage Error	±100μV

ACCURACY

Resolution	12 bits
Nonlinearity	Less than ±1/2 LSB
Differential Nonlinearity	Less than ±1/2 LSB
Inherent Quantizing Error	±1/2 LSB
System Accuracy	
Gain = 1	To within ±.03% FSR
Gain = 8	To within ±.05% FSR
Channel Cross Talk	-80dB at 1kHz
Gain Error	Adjustable to 0
Zero Error	Adjustable to 0
Noise	0.2 LSB rms

DYNAMIC PERFORMANCE

Channel Acquisition Time To within 1/2 LSB	2.5μs
A/D Conversion Time	2.5μs
Max. A/D Converter Throughput	200kHz (Throughput is defined as the maximum frequency of A/D conversions achievable by the board. The figure given above does not allow for software overhead necessary to retrieve the converted data.)
Sample And Hold Aperture Uncertainty	Less than 2ns
Sample And Hold Aperture Delay	40ns
Sample _Hold Feedthrough Attenuation	80dB at 1kHz
Sample And Hold Droop Rate	20μV/μs

THERMAL CHARACTERISTICS

A/D Zero Drift	±50µV/°C (unipolar); ±25ppm of FSR/°C (bipolar)
Amplifier Zero Drift	[±20µV/°C] + [(±10µV/°C)xGain]
Gain Drift	±40ppm of FSR/µC
Differential Linearity Drift	±3ppm of FSR/µC
Monotonicity	Monotonic, 0 to +50°C (32 to 122°F)

2.12 DT3377

ANALOG INPUTS

Number of Analog Inputs	4DI
Input Range	±10V (bipolar)
Gain Range	1 (fixed)
Output Data Codes	Two's complement
Input Impedance	"Off" channel: 100MΩ in parallel with 10pH; "On" channel: 100MΩ in parallel with 20pF
Bias Current	250±nA
Common Mode Input Voltage, Maximum	±10.5V
Common Mode Rejection Ratio (CMRR)	80dB at 1kHz
Maximum Input Voltage Without Damage, Power On	±27V
Maximum Input Voltage Without Damage, Power Off	±12V
Amplifier Input Noise	0.5 LSB rms
Channel-to-channel Input Voltage Error	±100μV

ACCURACY

Resolution	16 bits
Nonlinearity	0.003% of FSR
Differential Nonlinearity	0.0015% of FSR
Inherent Quantizing Error	±1/2 LSB
System Accuracy	To within ±.003% FSR
Channel Cross Talk	-100dB at 50kHz
Gain Error	Adjustable to 0
Zero Error	Adjustable to 0

DYNAMIC PERFORMANCE

Channel Acquisition Time To within 1/2 LSB	6µs
A/D Conversion Time	6µs
Max. A/D Converter Throughput	100kHz (Throughput is defined as the maximum frequency of A/D conversions achievable by the board. The figure given above does not allow for software overhead necessary to retrieve the converted data.)
Sample And Hold Aperture Uncertainty	0.2ns
Sample And Hold Aperture Delay	100ns
Sample And Hold Feedthrough Attenuation	94dB at 100kHz
Sample And Hold Droop Rate	7.5mV/ms

THERMAL CHARACTERISTICS

Offset Drift	±50µV/°C
Gain Drift	0.001% of FSR/µ0C
Differential Linearity Drift	±1.5ppm of FSR/°C
Monotonicity	Monotonic, 0 to +60°C (32 to 140°F)

2.13 DT3368-4SE AND DT3368-12SE

ANALOG INPUTS

Number of Analog Inputs	
DT3368-4SE	4SE
DT3368-12SE	12SE
Input Ranges	0 to +10V (unipolar) ±10V (bipolar)
Output Data Codes	Straight binary (unipolar) Offset binary (bipolar); Two's complement (bipolar); Jumper selectable: Sign extension is available with bipolar ranges
Gain	1 (fixed)
Input Impedance	"Off" channel: 10MΩ in parallel with 20pF; "On" channel: 10MΩ in parallel with 20pF
Bias Current	200nA
Maximum Input Voltage Without Damage, Power On	±31V
Maximum Input Voltage Without Damage, Power Off	±16V
Channel-to-Channel Input Voltage Error	±10mV

ACCURACY

Resolution	12 bits
Nonlinearity	Less than ±1/2 LSB
Differential Nonlinearity	±1/2 LSB
Inherent Quantizing Error	±1/2 LSB
System Accuracy	Within ±0.04% of FSR/°C
Single Channel Accuracy	Within ±0.03% of FSR/°C
Channel Crosstalk	-80dB at 50kHz
Gain Error	Adjustable to 0
Zero Error	Adjustable to 0

DYNAMIC PERFORMANCE

Channel Acquisition Time	6μs
A/D Conversion Time	4μs
System Aggregate Throughput of Channel 0	100kHz (Throughput is defined as the maximum frequency of A/D conversions achievable by the board. The figure given above does not allow for software overhead necessary to retrieve the converted data.)
Sample And Hold Aperture Uncertainty	±5ns
Sample And Hold Aperture Delay	100ns
Sample And Hold Feedthrough Attenuation	80dB
Sample And Hold Droop Rate	0.5mV/ms
Slewing Rate	5V/μs
Bandwidth	1MHz
Phase Match At 50kHz	Within .1°C

THERMAL CHARACTERISTICS

A/D Zero Drift Gain Tempco	±20μV/° (unipolar) ±20ppm of FSR/°C (bipolar)
Gain Drift	±30ppm of FSR/°C
Linearity Drift	±3ppm of FSR/°C
Monotonicity	Monotonic, 0 to +70°C (32 to 158°F)

2.14 EXTERNAL TRIGGER

Function	Initiate A/D conversions
Input Type	Edge sensitive; clocks on falling edge
Logic Compatibility	TTL
Origin	User device (such as DT2769 real-time clock)
Logic Load	Presents 1 TTL load
Logic High Input Voltage	2.4V minimum
Logic Low Input Voltage	0.8V maximum
Logic High Input Current	80μA @ 2.4V
Logic Low Input Current	2mA maximum
Minimum pulse width	
Clock High	200ns
Clock Low	200ns
Loading	Presents 1 TTL load to the user driver

2.15 INTERFACE CHARACTERISTICS

Compatible Bus	Q-bus
Interface Type	I/O mapped in 4K I/O page
Bus Loading	Board presents 1 dc load; 1 ac load
Number of Locations Occupied	8 word locations reserved
Base Address	Jumper-selectable over the range 760000 (oct) to 777760 (oct) in increments of 20 (oct)
Factory-assigned Base Address	771300 (oct)
Analog Data Format	
12-bit Models	Right-hand justified
Data Path	16 bits

2.16 POWER REQUIREMENTS

+5V	±5%, @ 2.5A, typical

2.17 PHYSICAL/ENVIRONMENTAL

Dimensions	8.39"H X 10.34"W X 0.43"D (21.5 X 26.5 X 1.1cm)
Weight	
Non-expanded Board	19.5 ounces (553g)
Expanded Board	24.0 ounces (680g)
Maximum Altitude	7,500 feet (2286 meters)
Operating Temperature Range	0 to +50°C (32 to 122°F)
Storage Temperature Range	-25 to +70 degrees C (-13 to 158°F)
Humidity	To 90%, non-condensing

2.18 CONNECTORS AND CABLES

Connector J1	26-pin header, male; mating connector: 26-pin header, female, 3M type, P/N 3399 or equivalent
Connector J2	50-pin header, male; mating connector: 50-pin header, female, 3M type, P/N 3425 or equivalent
Connector J3	20-pin header, male; mating connector: 20-pin header, female, 3M type, P/N 3421 or equivalent
Cable Assemblies	EP097, 26-conductor, flat ribbon;EP097, 8 ft; EP097-1, 1.5 ft; Connector 1: socket, P/N 3M-3399; Connector 2: socket, P/N 3M-3399
	EP035, 50-conductor, flat ribbon; EP035, 8 ft; Connector 1: socket, P/N 3M-3425; Connector 2: socket, P/N 3M-3425

	EP066, 50-conductor, flat twisted pair, 8 ft; Connector 1: socket, P/N 3M-3425; Connector 2: socket, P/N 3M-3425
EP164 Interconnection Panel	Used with expanded versions of the DT3362 series; converts three 20-pin inputs into one 50-pin output

2.19 COMPATIBLE SCREW TERMINAL PANELS

DT701-20	All models
DT701-50	Expanded models only; connected via J2 on DT3362 series
DT750/DT6700	All SE models
DT709-S	All SE models
DT709-Y	All SE models
DT756-Y and DT756-D	All SE models except DT3368-4SE and DT3368-12SE

CHAPTER 3

UNPACKING AND CONFIGURATION

3.1 INTRODUCTION

This chapter consists of two parts. The first part explains how to unpack the board. The second part discusses the configuration of user-selectable parameters which should be done prior to installing the board.

3.2 UNPACKING

Inspect the shipping carton immediately upon receipt for evidence of mishandling during transit. Depending on the condition of the shipping carton, proceed as outlined in the next two sections.

3.2.1 NO APPARENT DAMAGE

If no visible damage is found, proceed as follows:

1. Place the container on a stable surface and cut the shipping tape.

2. Grasp the front of the flat carton underneath and lift forward and up to open the top.

3. While placing one hand firmly on a metal ground, remove the board from the styrofoam chips. This will prevent any damage to the board components from possible static electricity build-up during transit.

4. After allowing a moment for the discharge of any static electricity into the metal ground, carefully unwrap the board or component from the anti-static bag.

5. Examine the board again for any possible damage. If any sign of damage is encountered, proceed as detailed in the next section.

3.2.2 VISIBLE DAMAGE

If the board container shows any visible damage, open it in the presence of the delivering carrier's agent. Then proceed to unpack it from the shipping container as detailed in the previous section. Then:

1. Examine the board for shipping damage. If damage is detected, do NOT perform any repair or installation of the damaged board. If the delivery carrier's agent was not present during unpacking, call for an insurance claim inspection.

2. Have the insurance agent inspect the damage and arrive at an agreement to repair or replace the damaged board.

3. If the board is to be repaired, call or write:

   Customer Service Department
   Data Translation, Inc.
   100 Locke Drive
   Marlboro, MA 01752-1192
   Tel: (617) 481-3700
   Telex: 951646 (DATATRANS MARO)
   Easylink 62825999

4. Request a Return Material Authorization (RMA) number. This must be obtained before any board will be accepted for return.

5. Repackage the damaged board in the anti-static wrapping and place it in its original shipping material or a secure container for shipment back to the factory.

   NOTE

   Ensure that the board is wrapped in anti-static (electrically conductive) packaging, and that it is handled with ground protection. Static electricity is potentially damaging to electrical components on Data Translation products.

6. Return the damaged board, with the RMA number, to the address above.

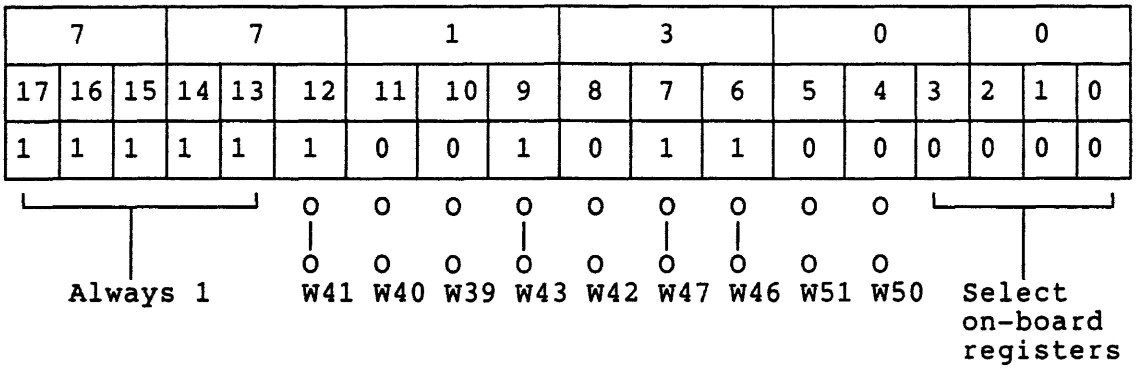
3.3 CONFIGURATION

Certain operational parameters on the DT3362 series boards are user-selectable and are configured via the addition or removal of jumpers. Jumpers are installed by connecting two jumper posts with a jumper plug or a wire wrap.

In this manual and on the DT3362 series board, jumpers are designated by either a capital or a lower case "w", followed by a number. A single capital "W" number--as W14 or W18--designates two adjacent posts which make up the jumper. Installing W14, for instance, consists of connecting together the pair of posts designated by the single number. A lower case "w" followed by a number designates a jumper post. In this case, the jumper is fully designated with two lower case "w" numbers such as w14 to w15. This type of designation is used when a jumper post is shared by two or more jumpers. Installing the jumper "w14 to w15 consists of connecting together the two separately labeled posts. Appendix C shows an assembly drawing which identifies the locations of all user accessible jumper.

3.3.1 DEVICE BASE ADDRESS SELECTION

The DT3362 has been designed such that its base address (the I/O address associated with the ADCSR; see Chapter 5) can be assigned any value from 760000 to 777760 (octal) in the I/O address range of the processor, in increments of 20 (octal). The addresses of the other registers are fixed relative to the location of the ADCSR. Figure 3-1 shows the factory-set value of 771300 for the base address implemented on the board.



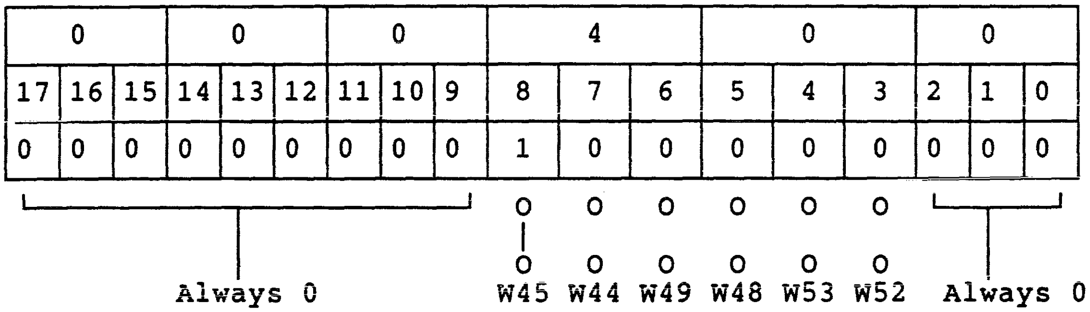
7			7			1			3			0			0		
17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	0	0	1	0	1	1	0	0	0	0	0	0

FIGURE 3-1: BASE ADDRESS SELECTION OF 771300 (OCTAL)

As shown in Figure 3-1, for an address bit, a value of 1 is selected by installing the corresponding jumper; a value of 0 is selected by removing the jumper. Thus W41, W43, W47, and W46 must be installed as shown to select a value of 1 for bits 12, 9, 7, and 6, respectively.

3.3.2 DEVICE INTERRUPT VECTOR ADDRESS SELECTION

The DT3362 series board can send two unique interrupts to the processor. These interrupts can occur on the A/D Done/DMA Done condition and on the A/D Error condition. The interrupt vector associated with the A/D Done or DMA Done condition can be assigned any value in the 000-770 (octal) range in increments of 10 (octal). The vector address associated with the A/D Error condition is four locations higher. Thus to set an address of 400 bit 8 must be set requiring the installation of jumper W45. The interrupt vector of the error condition would then be 400 + 04 = 404. Figure 3-2 shows how the factory default value of 400 (octal) is set for the A/D Done/DMA Done interrupt.



0			0			0			4			0			0		
17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

FIGURE 3-2: INTERRUPT VECTOR ADDRESS SELECTION OF 400 (OCTAL)

3.4 A/D CONFIGURATION JUMPERING

This section describes how the user can select various analog configurations for the DT3362 series board.

3.4.1 INPUT MODE

Not all DT3362 series models offer selection of the input mode. Table 3-1 lists all the models and indicates those with user-selectable input modes.

TABLE 3-1: DT3362 SERIES INPUT MODES

A/D BOARD	USER-SELECTABLE	FACTORY-SHIPPED CONFIGURATION
DT3362-16SE/8DI	Yes	Differential
DT3362-64SE/32DI	Yes	Differential
DT3362-16SE/8DI-PGH	Yes	Differential
DT3362-64SE/32DI-PGH	Yes	Differential
DT3362-F-16SE	No	Single-ended
DT3362-F-64SE	No	Single-ended
DT3362-F-8DI	No	Differential
DT3362-F-32DI	No	Differential
DT3362-G-16SE	No	Single-ended
DT3362-G-64SE	No	Single-ended
DT3362-G-8DI	No	Differential
DT3362-G-32DI	No	Differential
DT3362-H-16SE	No	Single-ended
DT3362-H-64SE	No	Single-ended
DT3362-H-8DI	No	Differential
DT3362-H-32DI	No	Differential
DT3362-H-16SE-PGH	No	Single-ended
DT3362-H-64SE-PGH	No	Single-ended
DT3362-H-8DI-PGH	No	Differential
DT3362-H-32DI-PGH	No	Differential
DT3377	No	Differential
DT3368-4SE	No	Single-ended
DT3368-12SE	No	Single-ended

Table 3-2 gives the setting of jumpers for the selection of single-ended or differential input mode on these boards. The input modes of the other boards in the series are configured at the factory and should not be tampered with.

TABLE 3-2: INPUT MODE SELECTION FOR
DT3362-16SE/8DI, DT3362-64SE/32DI,
DT3362-16SE/8DI-PGH, AND DT3362-64SE/32DI-PGH

INPUT MODE	w14 TO w15	w16 TO w18	w4 TO w5	w6 TO w7	w14 TO w16	w23 TO w24	w3 TO w4	w5 TO w6
Single-ended	In	In	In	In	Out	Out	Out	Out
Differential	Out	Out	Out	Out	In	In	In	In

NOTES:

1. Jumpers connecting to w4 and w6 need only be installed when the board has 64SE or 32DI channels. For boards with 16SE or 8DI channels connections to w4 and w6 are not required. Similarly, those connections are not made on the DT3377 and DT3368-4SE.

3.4.2 A/D INPUT RANGE

All DT3362 series boards except the DT3377 permit selection of unipolar or bipolar input ranges. Unipolar input ranges accept positive voltages only. They should be used with input levels ranging from 0 to +1.25V to 0 to +10V, depending on the board model and gain setting of the A/D converter's instrumentation amplifier. With differential inputs the high and low ends can be reversed for zero to negative value inputs.

Bipolar input ranges accept voltages which can be positive or negative. They should be used with input levels from ±1.25V to ±10V, again depending on the board model and gain setting of the instrumentation amplifier.

Table 3-3 shows that all models of the DT3362 series except the DT3377 have user-selectable input ranges. The factory-shipped configuration of the input range is bipolar on all models.

TABLE 3-3: DT3362 SERIES INPUT RANGES

A/D BOARD	USER-SELECTABLE	FACTORY-SHIPPED CONFIGURATION
DT3362-16SE/8DI	Yes	Bipolar
DT3362-64SE/32DI	Yes	Bipolar
DT3362-16SE/8DI-PGH	Yes	Bipolar
DT3362-64SE/32DI-PGH	Yes	Bipolar
DT3362-F-16SE	Yes	Bipolar
DT3362-F-64SE	Yes	Bipolar
DT3362-F-8DI	Yes	Bipolar
DT3362-F-32DI	Yes	Bipolar
DT3362-G-16SE	Yes	Bipolar
DT3362-G-64SE	Yes	Bipolar
DT3362-G-8DI	Yes	Bipolar
DT3362-G-32DI	Yes	Bipolar
DT3362-H-16SE	Yes	Bipolar
DT3362-H-64SE	Yes	Bipolar
DT3362-H-8DI	Yes	Bipolar
DT3362-H-32DI	Yes	Bipolar
DT3362-H-16SE-PGH	Yes	Bipolar
DT3362-H-64SE-PGH	Yes	Bipolar
DT3362-H-8DI-PGH	Yes	Bipolar
DT3362-H-32DI-PGH	Yes	Bipolar
DT3377	No	Bipolar
DT3368-4SE	Yes	Bipolar
DT3368-12SE	Yes	Bipolar

Table 3-4 indicates which jumpers on the DT3362 series should be installed and removed to set the board for unipolar or bipolar operation.

TABLE 3-4: INPUT RANGE SELECTION

A. DT3362-16SE/8DI, DT3362-64SE/32DI,
DT3362-16SE/8DI-PGH,
DT3362-64SE/32DI-PGH.

INPUT RANGE	W26	W13	W12
Unipolar	Out	In	In
Bipolar (FC)	In	In	Out

B. DT3362-F-16SE, DT3362-F-64SE,
DT3362-F-8DI, DT3362-F-32DI,
DT3362-G-16SE, DT3362-G-64SE,
DT3362-G-8DI, DT3362-G-32DI,
DT3362-H-16SE, DT3362-H-64SE,
DT3362-H-8DI, DT3362-H-32DI,
DT3362-H-16SE-PGH, DT3362-H-64SE-PGH
DT3362-H-8DI-PGH, DT3362-H-32DI-PGH

INPUT RANGE	W26	W13	W12
Unipolar	Out	In	Out
Bipolar (FC)	In	Out	Out

C. DT3368-4SE and DT3368-12SE.

INPUT RANGE	W26	W13	W12
Unipolar	Out	In	In
Bipolar (FC)	In	In	Out

NOTES:

1. The input range of the DT3377 is bipolar and not user-selectable.
2. FC is factory configuration.
3. If the programmable gain is used, the actual input range will be the indicated range divided by the gain.

NOTE

The A/D module on the D3362 series board should be recalibrated whenever the input polarity is changed.

3.4.3 SELECTION OF OUTPUT CODING

At the end of an A/D conversion, the incoming analog input signal has been changed into an equivalent digital output which permits analog input information to be understood and manipulated by the Q-bus master. The output of the DT3362 series board is a binary data word, the coding of which depends on the converter module and board jumpering.

With unipolar input ranges straight binary output code should be used. Digital outputs for bipolar input ranges can be represented using either offset binary or two's complement binary coding.

Table 3-5 shows which codes are available on each model, and which is the factory-shipped configuration.

TABLE 3-5: DT3362 SERIES OUTPUT CODES

A/D BOARD (1)	OUTPUT CODE
DT3362-16SE/8DI DT3362-64SE/32DI	Straight Binary Offset Binary Two's Complement (FC)
DT3362-16SE/8DI-PGH DT3362-64SE/32DI-PGH	Straight Binary Offset Binary Two's Complement (FC)
DT3362-F-16SE DT3362-F-64SE DT3362-F-8DI DT3362-F-32DI	Straight Binary Offset Binary Two's Complement (FC)
DT3362-G-16SE DT3362-G-64SE DT3362-G-8DI DT3362-G-32DI	Straight Binary Offset Binary Two's Complement (FC)
DT3362-H-16SE DT3362-H-64SE DT3362-H-8DI DT3362-H-32DI	Straight Binary Offset Binary Two's Complement (FC)
DT3362-H-16SE-PGH DT3362-H-64SE-PGH DT3362-H-8DI-PGH DT3362-H-32DI-PGH	Straight Binary Offset Binary Two's Complement (FC)
DT3377	Two's Complement
DT3368-4SE DT3368-12SE	Straight Binary Offset Binary Two's Complement (FC)

NOTE:
1. FC is factory configuration.

Table 3-6 indicates the jumper configuration required to select straight binary, offset binary, or two's complement binary coding.

TABLE 3-6: SELECTION OF DATA CODING

CODING	W37	W38	W28	W27
Two's Complement (FC)	In	Out	Out	In
Binary	Out	In	In	Out
Offset Binary	Out	In	In	Out

NOTES:
1. FC is factory configuration.
2. Output coding is two's complement for DT3377.

NOTE

The output coding and the input polarity (see previous section) selected on the A/D board should be mutually compatible. Offset binary or two's complement binary output coding should be selected for bipolar inputs, and straight binary output coding for unipolar inputs.

3.4.3.1 Input Gain And A/D Range

The input range which the module will accept depends on the gain setting of the programmable gain amplifier. It can thus extend from 0 to +1.25V to 0 to +10V for unipolar ranges, and from ±1.25V to ±10V for bipolar ranges. The actual input voltage range, however, depends on the resolution and the gain of the instrumentation amplifier. The actual full scale voltage range is the lowest voltage which will produce the highest digital value in the selected output code. For all models of the DT3362 series except the DT3377 the output code is 12 bits wide. The maximum output of the A/D converters on these boards is all ones in straight binary or offset binary, and zero followed by all ones in two's complement. This output is zero followed by all one's for the DT3377 which is set for two's complement output coding. In all cases, the maximum output corresponds to a reading of full scale voltage minus 1 LSB (least significant bit). Increasing the input voltage beyond the actual full scale input voltage will not result in a change to the digital code output from the A/D converter. Table 3-7 gives the codes for full scale outputs in three notations for all DT3362 series boards.

TABLE 3-7: FULL SCALE OUTPUT CODES IN VARIOUS NOTATIONS

OUTPUT NOTATION	FULL SCALE OUTPUT CODE BINARY	OCTAL
Straight Binary (12-Bits)	0000 1111 1111 1111	007777
Offset Binary (12-Bits)	0000 1111 1111 1111	007777
Two's Complement (12-Bits)	0000 0111 1111 1111	003777
Two's Complement (16-Bits)	0111 1111 1111 1111	077777

Table 3-8 gives the actual full scale inputs for the various gains at 12-bit and 16-bit resolutions.

TABLE 3-8: FULL SCALE RANGES FOR
HIGH LEVEL INPUT BOARDS

GAIN	INPUT RANGE	ACTUAL FULL SCALE INPUT 12 BITS	 16 BITS
	Unipolar		
	0 to +10 V	+9.9976 V	Not
2	0 to + 5 V	+4.9988 V	
4	0 to + 2.5 V	+2.4994 V	Applicable
8	0 to + 1.25V	+1.2497 V	
	Bipolar		
1	±10 V	+9.9951 V	+9.99970 V
2	± 5 V	+4.9976 V	+4.99985 V
4	± 2.5 V	+2.4988 V	+2.49992 V
8	± 1.25 V	+1.2494 V	+1.24996 V

3.4.4 JUMPER CONFIGURATIONS ON 22-BIT DMA ADDRESS BOARD

The 22-bit DMA addressing is implemented on Revision F and higher revision boards. Unless otherwise noted, the discussions in sections 3.4.5.1, 3.4.5.2, and 3.4.5.3. pertain to those boards only.

3.4.4.1 Selection Of DMA Wrap Mode

The wrap mode refers to the operation of the six most significant bits of the 22-bit DMA current address. In the wrap mode these bits are static and the DMA "wraps" to the beginning of the 64Kbyte segment when the lower bits overflow. In the non-wrap mode, the upper bits are dynamic and the overflow from the lower 16 bits causes the upper six bits to increment. The wrap mode is selected by removing W58, the non-wrap mode is selected by installing W58 as shown in Table 3-9.

NOTE

> To avoid possible complications, it is recommended that the first and last 64Kbyte segments of memory not be used. The first segment contains the vectors and the stack, and the last segment contains the I/O page, whereas all locations are available on all other memory segments.

TABLE 3-9: DMA WRAP MODE SELECTION

DMA ADDRESS MODE	JUMPER W58
Wrap	Out
No-wrap (FC)	In

NOTE:
1. FC is factory configuration.

3.4.4.1.1 Selection Of 22-bit DMA Addressing

The 22-bit addressing feature of the Revision F and higher revision boards is selected by configuring jumpers W59 and W60 as shown in Table 3-10.

TABLE 3-10: SELECTION OF DMA ADDRESSING

	JUMPERS	
DMA ADDRESSING	W59	W60
22-bit	In	Out
18-bit (FC)	Out	In

3.4.4.2 Triggers

In most applications, the trigger required by the A/D converters on the DT3362 series boards will be provided by the Data Translation DT2769 clock board. With the standard versions of the A/D boards, cable EP140 is used to connect jumper post w8 on the DT3362 series with the CLK OVFL pin on the DT2769.

When the LDT versions of the A/D boards (LDT3362 and LDT3368) are used, jumper post w8 must be connected to jumper post w9, and the user must connect the CLK OVFL BNC connector on the DT2769 front panel with the EXT TRIG L BNC connector on the DT3362 series board's front panel.

To use an external clock source, connect jumper post w8 to w9. The external trigger should be connected to pin 19 of connector J3. The section on the external trigger in Chapter 2 gives complete electrical specifications for the external trigger input.

3.4.4.3 Reference Jumpers

Table 3-11 gives a list of jumpers on the etch Revision F and later boards which are set in the factory and are not user-configurable.

TABLE 3-11: REFERENCE JUMPERS
FOR REVISION F AND LATER BOARDS

JUMPER	CONFIGURATION
W1*	In
W2*	Out
W54	In
W55	Out
W56	In
W57	Out

NOTE:
1. The asterisked jumpers are present on etch revision E and F boards.

A set of jumpers are configured in the factory for all models of the DT3362 series and are not configurable by the user. The configurations of these jumpers are given in Table 3-12. See Appendix A for a complete list of all jumpers and their configurations on the DT3362 series board.

TABLE 3-12: REFERENCE JUMPERS FOR ALL MODELS

REFERENCE JUMPER	CONFIGURATION
W25	Out
W29	In
W30	In
W31	In
W32	Out
W33	In
W34	Out
W35	Out
W36	Out

CHAPTER 4

SYSTEM INTERCONNECTIONS

4.1 INTRODUCTION

This chapter describes the connections of the DT3362 series board to the processing host computer system and data acquisition accessories. The connection to the host computer system is made by plugging directly into the edge connector of the Q-bus computer's backplane. Connection to the data acquisition accessories--screw terminal/signal conditioning panels--is made with appropriate cable assemblies.

4.2 ANALOG INPUT CONNECTIONS

All connections between the DT3362 series board and the analog signal input points such as barrier terminals on a screw terminal panel are made using connectors J3 and J2. J3 is a 20-pin header which connects the board to a compatible data acquisition accessory. J2 is a 50-pin header which is present on the expanded versions of the DT3362 series and performs the same function as J3. Basic versions of the DT3362 series use only connector J3 to connect to analog input channels, while the expanded versions use both J3 and J2 connectors.

4.2.1 J3 AND J2 CONNECTIONS ON DT3362- PREFIXED MODELS

Tables 4-1 and 4-2 show the pin assignments of the J3 and J2 connectors on models of the DT3362 series prefixed with DT3362-.

TABLE 4-1: J3 PIN ASSIGNMENTS
ON ALL DT3362- PREFIXED MODELS

SIGNAL NAME	PIN		SIGNAL NAME
CH0	1	2	CH8/RET0
CH1	3	4	CH9/RET1
CH2	5	6	CH10/RET2
CH3	7	8	CH11/RET3
CH4	9	10	CH12/RET4
CH5	11	12	CH13/RET5
CH6	13	14	CH14/RET6
CH7	15	16	CH15/RET7
A GND	17	18	AMP IN
EXT. TRIG L	19	20	D GND

NOTES:

1. The channel numbers listed first are single-ended. The differential channels are listed second.
2. Tie all unused analog inputs to A GND.

TABLE 4-2: J2 EXPANDER PIN ASSIGNMENTS
ON DT3362- PREFIXED MODELS

SIGNAL NAME	PIN		SIGNAL NAME
CH24/RET8	1	2	CH16/CH8
CH25/RET9	3	4	CH17/CH9
CH26/RET10	5	6	CH18/CH10
CH27/RET11	7	8	CH19/CH11
CH28/RET12	9	10	CH20/CH12
CH29/RET13	11	12	CH21/CH13
CH30/RET14	13	14	CH22/CH14
CH31/RET15	15	16	CH23/CH15
CH40/RET16	17	18	CH32/CH16
CH41/RET17	19	20	CH33/CH17
CH42/RET18	21	22	CH34/CH18
CH43/RET19	23	24	CH35/CH19
CH44/RET20	25	26	CH36/CH20
CH45/RET21	27	28	CH37/CH21
CH46/RET22	29	30	CH38/CH22
CH47/RET23	31	32	CH39/CH23
CH56/RET24	33	34	CH48/CH24
CH57/RET25	35	36	CH49/CH25
CH58/RET26	37	38	CH50/CH26
CH59/RET27	39	40	CH51/CH27
CH60/RET28	41	42	CH52/CH28
CH61/RET29	43	44	CH53/CH29
CH62/RET30	45	46	CH54/CH30
CH63/RET31	47	48	CH55/CH31
A GND	49	50	A GND

NOTES:

1. The channel numbers listed first are single-ended. The differential channels are listed second.
2. Tie all unused analog inputs to A GND.

4.2.2 J3 CONNECTION ON DT3377

Table 4-3 shows the pin assignments of the J3 connector on the DT3377 board.

TABLE 4-3: J3 PIN ASSIGNMENTS ON DT3377

SIGNAL NAME	PIN	PIN	SIGNAL NAME
CH0	1	2	CH0 Return
NC	3	4	NC
CH1	5	6	CH1 Return
NC	7	8	NC
CH2	9	10	CH2 Return
NC	11	12	NC
CH3	13	14	CH3 Return
NC	15	16	NC
A GND	17	18	NC
EXT. TRIG L	19	20	D GND

NOTES:
1. NC = No Connection.
2. Tie all unused analog inputs to A GND.

4.2.3 J3 AND J2 CONNECTIONS ON DT3368- PREFIXED MODELS

Tables 4-4 and 4-5 show the pin assignments of the J3 and J2 connectors on the basic and expanded versions of the DT3368-4SE and DT3368-12SE models.

TABLE 4-4: J3 PIN ASSIGNMENTS ON
DT3368-4SE AND DT3368-12SE

SIGNAL NAME	PIN	PIN	SIGNAL NAME
CH0	1	2	A GND
NC	3	4	NC
CH1	5	6	A GND
NC	7	8	NC
CH2	9	10	A GND
NC	11	12	NC
CH3	13	14	A GND
NC	15	16	NC
A GND	17	18	NC
EXT. TRIG L	19	20	D GND

NOTES:
1. NC = No Connection.
2. Tie all unused analog inputs to A GND.

TABLE 4-5: J2 EXPANDER PIN ASSIGNMENTS
ON DT3368-12SE

SIGNAL NAME	PIN		SIGNAL NAME
A GND	1	2	CH 4
NC	3	4	NC
NC	5	6	NC
A GND	7	8	CH 5
NC	9	10	NC
NC	11	12	NC
A GND	13	14	CH 6
NC	15	16	NC
NC	17	18	NC
A GND	19	20	CH 7
NC	21	22	NC
NC	23	24	NC
A GND	25	26	CH 8
NC	27	28	NC
NC	29	30	NC
A GND	31	32	CH 9
NC	33	34	NC
NC	35	36	NC
A GND	37	38	CH 10
NC	39	40	NC
NC	41	42	NC
A GND	43	44	CH 11
NC	45	46	NC
NC	47	48	NC
A GND	49	50	A GND

NOTES:
1. NC = No Connection.
2. Tie all unused analog inputs to A GND.

4.3 EXTERNAL PORT CONNECTION

The DT3362 series board uses the 26-pin header J1 connector to mate with the DT3369 dual port memory device via the External Port. The pin assignments of the J1 connector are given in Table 4-6.

TABLE 4-6: EXTERNAL PORT CONNECTOR J1
PIN ASSIGNMENTS

SIGNAL NAME	PIN		SIGNAL NAME
MEM DAT 0	1	2	MEM DAT 1
MEM DAT 2	3	4	MEM DAT 3
MEM DAT 4	5	6	MEM DAT 5
MEM DAT 6	7	8	MEM DAT 7
MEM DAT 8	9	10	MEM DAT 9
MEM DAT 10	11	12	MEM DAT 11
MEM DAT 12	13	14	MEM DAT 13
MEM DAT 14	15	16	MEM DAT 15
MEM REPLY L	17	18	D GND
WRITE L	19	20	D GND
READ L	21	22	D GND
LAST TRANS L	23	24	D GND
READY L	25	26	D GND

4.4 EXTERNAL TRIGGER INPUT

The external trigger input allows the user to synchronize conversions to an external clock or event. The input requires a TTL-compatible signal and initiates conversions on the high-to-low transition of the signal. The duty cycle of the signal is not critical.

The Data Translation DT2769 Real-Time Clock is a possible source of trigger inputs. This is a software programmable clock for the Q-bus whose output can be used to time precise intervals between conversions. Refer to Section 3.4.6, "Using an External Trigger", for the jumper configuration required when using the external trigger.

4.5 CONNECTIONS TO DATA ACQUISITION ACCESSORIES

Connections to compatible data acquisition accessories are made via the J3 connector for the non-expanded boards, and via J2 and J3 connectors on the expanded versions. The A/D expander accommodates three compatible screw terminal panels. The connection to the expander requires the EP164 interconnection panel.

The screw terminal panels compatible with the DT3362 series include the following:

1. DT701-20
2. DT701-50
3. DT709 series
4. DT756 series
5. DT750/DT6700 series

Figures 4-1 through 4-6 illustrate the connections of the compatible screw terminal panels to the DT3362 series boards.

Figure 4-1 shows the connection of the DT701-20 and DT701-50 to the DT3362 series boards.

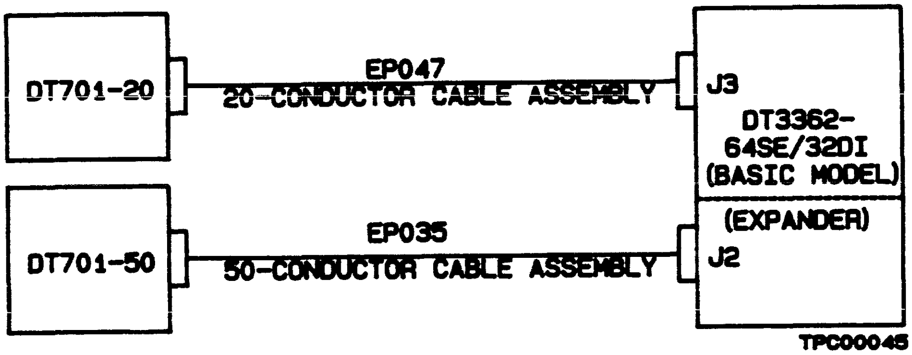


FIGURE 4-1: CONNECTION OF DT701-20 AND DT701-50 TO DT3362 SERIES BOARDS

Figure 4-2 shows the connection of the DT709 series and the DT6700 series screw terminal panels to single-ended versions of the DT3362 series boards.

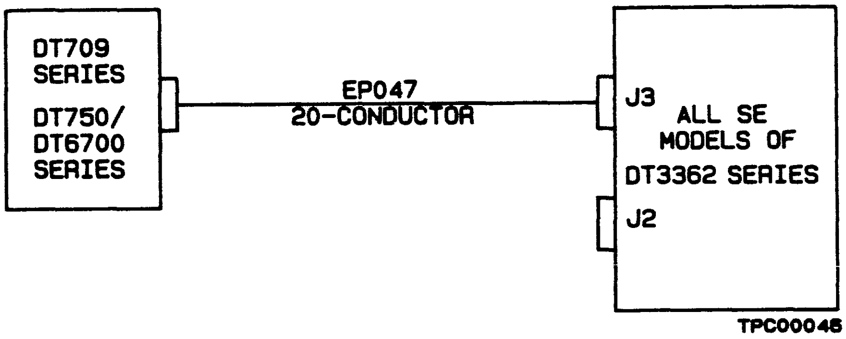


FIGURE 4-2: CONNECTION OF DT709 SERIES AND DT6700 SERIES TO SE VERSIONS OF THE DT3362 SERIES BOARDS

Figure 4-3 shows the connection of the DT756 series to all single-ended versions of the DT3362 series except the DT3368-4SE and DT3368-12SE.

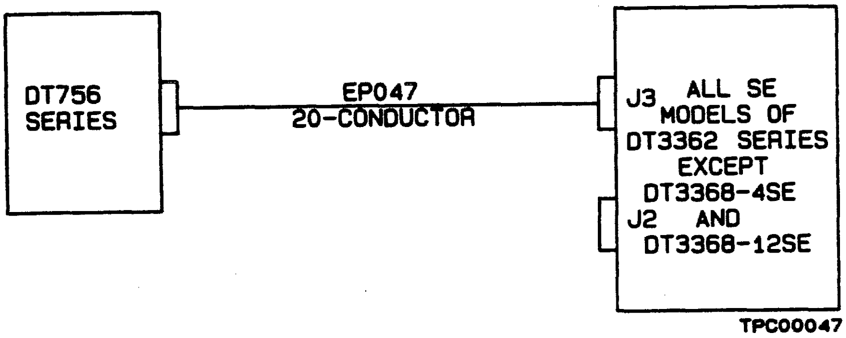


FIGURE 4-3: CONNECTION OF DT756 SERIES WITH SE VERSIONS OF THE DT3362 SERIES EXCEPT DT3368-4SE AND DT3368-12SE

Figure 4-4 shows the connection of three screw terminal panels to the A/D expander of the DT3362 series via the EP164 interconnection panel.

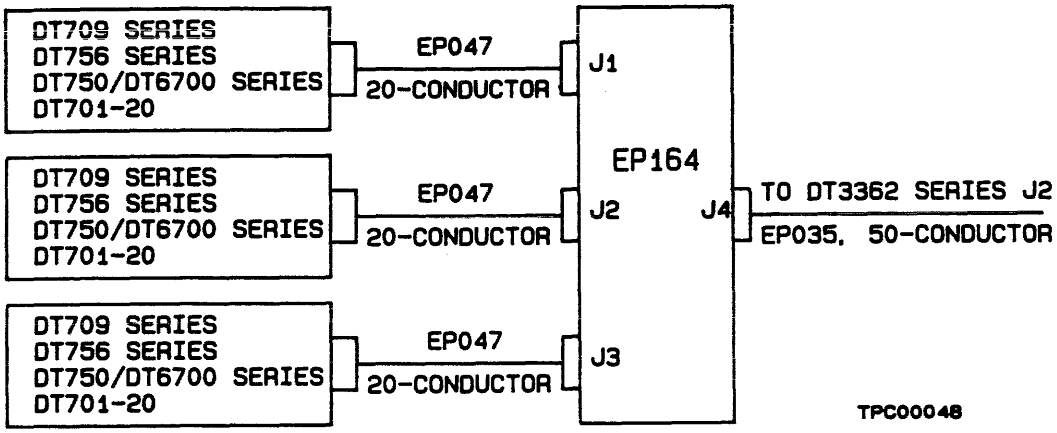


FIGURE 4-4: CONNECTION OF SCREW TERMINAL PANELS
TO THE A/D EXPANDER OF THE DT3362 SERIES

4.6 RECOMMENDED ANALOG INPUT CONNECTION SCHEMES

This section describes analog input connection schemes to the DT3362 series A/D boards. Each connection scheme is discussed separately, with a diagram representing the connection. Input modes possible with each model of the DT3362 series are indicated in Table 4-7.

TABLE 4-7: DT3362 SERIES INPUT MODES

	INPUT MODE		
MODEL	SE	PSEUDO-DI	DI
DT3362-8DI/16SE	Yes	Yes	Yes
DT3362-32DI/64SE	Yes	Yes	Yes
DT3362-8DI/16SE-PGH	Yes	Yes	Yes
DT3362-32DI/64SE-PGH	Yes	Yes	Yes
DT3362-F-8DI	No	No	Yes
DT3362-F-32DI	No	No	Yes
DT3362-F-16SE	Yes	Yes	No
DT3362-F-64SE	Yes	Yes	No
DT3362-G-8DI	No	No	Yes
DT3362-G-32DI	No	No	Yes
DT3362-G-16SE	Yes	Yes	No
DT3362-G-64SE	Yes	Yes	No
DT3362-H-8DI	No	No	Yes
DT3362-H-32DI	No	No	Yes
DT3362-H-16SE	Yes	Yes	No
DT3362-H-64SE	Yes	Yes	No
DT3362-H-8DI-PGH	No	No	Yes
DT3362-H-32DI-PGH	No	No	Yes
DT3362-H-16SE-PGH	Yes	Yes	No
DT3362-H-64SE-PGH	Yes	Yes	No
DT3377	No	No	Yes
DT3368-4SE	Yes	No	No
DT3368-12SE	Yes	No	No

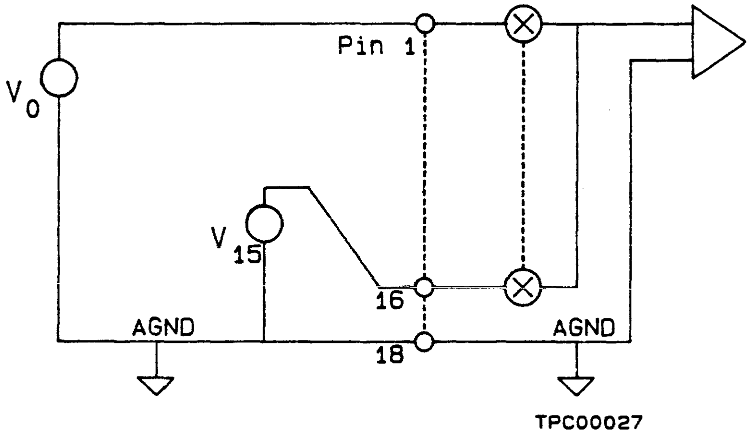
4.6.1 SINGLE-ENDED INPUTS

In single-ended connections, input signals have a common low side which is connected to the Analog Ground (AGND, pin 17 of J3, or pins 49 and 50 of J2). This connection scheme gives you maximum channel density. The major disadvantage of the single-ended connection scheme is that the instrumentation amplifier does not provide common mode rejection of the input signal, as it does in differential input connections. The single-ended connection scheme is available on models DT3362-16SE/8DI, DT3362-64SE/32DI, DT3362-16SE/8DI-PGH, DT3362-64SE/32DI-PGH, DT3362-F-16SE, DT3362-F-64SE, DT3362-G-16SE, DT3362-G-64SE, DT3362-H-16SE, DT3362-H-64SE, DT3362-H-16SE-PGH, DT3362-H-64SE-PGH, DT3368-4SE, and DT3368-12SE. Figure 4-5 shows how to connect single-ended inputs to these boards.

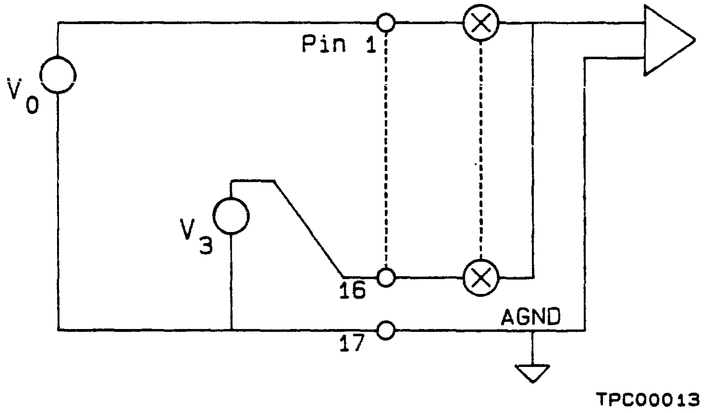
NOTE

> When operating in this input scheme, the user should always reference Amp Lo (AMP IN) to Analog Ground. This is accomplished by connecting Amp Lo to Analog Ground (pin 18 to pin 17 of the J3 connector). Failure to do so will result in inaccurate data conversions. In expanded versions, Amp Lo is tied only on the J3 connector. The DT3368-4SE and DT3368-12SE have no Amp Lo connection. In both cases the return point of each channel should be tied directly to Analog Ground.

Single-ended inputs should be restricted to short lead lengths (less than 15 feet) and gains less than 10.



ALL MODELS EXCEPT DT3368-4SE AND DT3368-12SE



MODELS DT3368-4SE AND DT3368-12SE

NOTES:

1. All signal inputs must be referenced to Analog Ground.
2. All unused analog input channels should be connected to Analog Ground. Failure to do so can result in inaccurate A/D conversions on all other channels, those closest to the floating channel(s) being affected most.

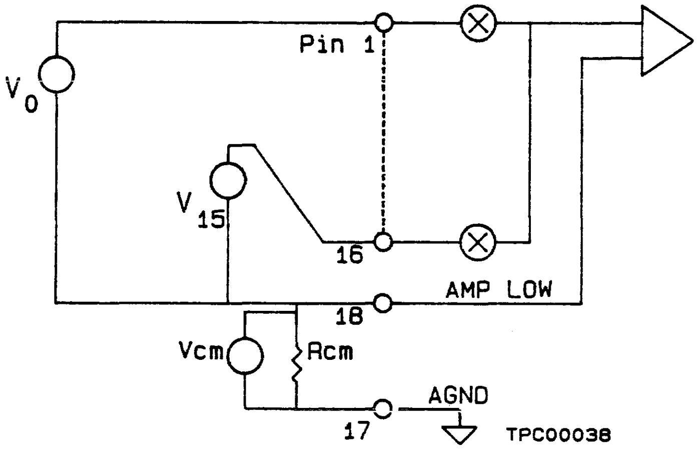
FIGURE 4-5: SINGLE-ENDED INPUT CONNECTION

4.6.2 PSEUDO-DIFFERENTIAL INPUTS

The pseudo-differential input is a variation of the single-ended input which provides the user with some degree of common mode noise rejection without sacrificing the number of input channels. To implement this input mode, the user jumpers the board for single-ended inputs and ties all the return sides of the single-ended inputs to the low side of the instrumentation amplifier (Amp Lo). This is possible if the Amp Lo connection is brought out to the user input connector, such as a screw terminal panel. Amp Lo in turn is connected externally to Analog Ground via 1 to 10 kilohm resistor. In this manner the input instrumentation amplifier can reject the common mode noise. Figure 4-6 shows how to connect the Amp Lo (AMP IN) and AGND pin for pseudo-differential operation.

NOTE

> The pseudo-differential input scheme can be applied only to channels served by the J3 connector on DT3362-16SE/8DI, DT3362-64SE/32DI, DT3362-16SE/8DI-PGH, DT3362-64SE/32DI-PGH, DT3362-F-16SE, DT3362-F-64SE, DT3362-G-16SE, DT3362-G-64SE, DT3362-H-16SE, DT3362-H-64SE, DT3362-H-16SE-PGH and DT3362-H-64SE-PGH. It is typically used with lead lengths not exceeding 25 feet and gains no more than 10. Channels served by the J2 connector cannot accommodate pseudo-differential connections since the expansion module does not provide an Amp Lo pin for connection of the return lines. For the same reason, the DT3368-4SE and DT3368-12SE do not accommodate pseudo-differential connection schemes.



NOTES:

1. All signal inputs must be referenced to Analog Ground.
2. All unused analog input channels should be connected to Analog Ground. Failure to do so can result in inaccurate A/D conversions on all other channels, those closest to the floating channel(s) being affected most.
3. Rcm is user-supplied bias return resistor. Its value is typically between 1 and 10 kilohms.

FIGURE 4-6: PSEUDO-DIFFERENTIAL INPUT CONNECTION

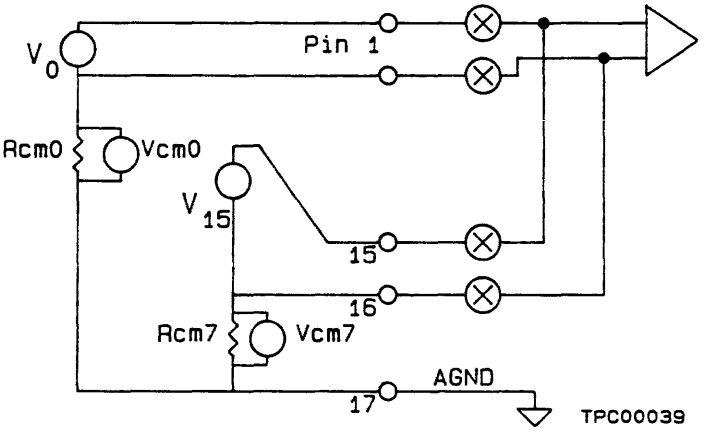
4.6.3 DIFFERENTIAL INPUTS

In a differential input scheme, two multiplexer switches are used on each channel. Thus the number of channels which can be accommodated is cut in half. The benefits are that the common mode voltages, that is, voltages common to both the high and low side of an analog input, can be rejected by the differential input instrumentation amplifier. This common mode rejection results in a much quieter system. This input configuration can be used with input ranges as low as 10mV full scale with twisted pair (low level) shielded input cables. Lead lengths may suit user requirements but should be kept as short as possible.

NOTE

> The low end of each differential input must be referenced to Analog Ground. This can be done by connecting a 1 to 10 kilohm resistor between the low end of each differential input and Analog Ground. If all inputs share a common ground, a single 10 kilohm resistor can be connected between the input signal common and the board Analog Ground.

The DT3362 series boards which can be operated differentially are the following: DT3362-16SE/8DI, DT3362-64SE/32DI, DT3362-16SE/8DI-PGH, DT3362-64SE/32DI-PGH, DT3362-F-8DI, DT3362-F-32DI, DT3362-G-8DI, DT3362-G-32DI, and DT3377. Figure 4-7 shows how to connect a differential input.



NOTES:

1. All signal inputs must be referenced to Analog Ground.
2. All unused analog input channels should be connected to Analog Ground. Failure to do so can result in inaccurate A/D conversions on all other channels, those closest to the floating channel(s) being affected most.
3. Rcm is user-supplied bias return resistor. Its value is typically between 1 and 10 kilohm.

FIGURE 4-7: DIFFERENTIAL INPUT CONNECTION

4.7 CONNECTION GUIDELINES

To optimize the performance of a system, certain guidelines must be considered in connecting analog signals to the A/D board. Following the guidelines and observing some precautions will minimize the pickup of electrical noise.

4.7.1 TWISTED PAIR INPUT LINES

The effects of magnetic coupling on the input signals can be reduced in differential input schemes by twisting the input lines of the signal. This is effective since the induced voltages on the two lines are of opposite polarity and tend to match, canceling out any common voltage. This cable type should not be used with single-ended inputs.

4.7.2 SHIELDED INPUT LINES

The effects of electrostatic coupling can be reduced by shielding the input lines. This becomes important if the source has a high impedance. The shield should only be tied to ground at the instrument end. This will prevent ground loop currents.

4.7.3 INPUT SETTLING WITH HIGH SOURCE IMPEDANCE

Solid state multiplexers inject a small amount of charge into the input lines when channels are switched. This can cause a transient error due to the input source impedance time constants. All Data Translation A/D boards allow for input settling upon new channel selection. The settling time varies for the different systems available and is controlled by the timing capacitor.

Normally, the control logic allows sufficient time for the injected charge to settle to less than 1/2 LSB of error. However, more time may be needed when the multiplexer is switching an input channel with high source impedance, particularly when large amounts of shunt capacitance exists in the interconnection cables. On 50kHz boards, for example, source impedance/cable shunt capacitance products greater than 1 microsecond (1 kilohm-1000pF) should be avoided if less than 1/2 LSB error is desired. Assuming a twisted pair cable capacitance of 50pF/foot and 1 kilohm source impedance, this condition restricts the cable length to a maximum of twenty feet on 50kHz models. Note also that settling error can be minimized by increasing the internal time-out with an external timing capacitor Ct (about 60pF/μs).

CHAPTER 5

ARCHITECTURE

5.1 INTRODUCTION

The high performance of the DT3362 series board is made possible by certain structural and operational features built into its architecture. These include the registers, multiplexer channel-list, external port for external dual port memory access, and DMA transfer methods.

5.2 REGISTERS

The registers of the DT3362 series data acquisition systems are designed to meet the requirements of standard DEC Q-bus interfaces. They are structured around a Control and Status Register (see following section) for complete software control of the A/D conversion operations. Table 5-1 summarizes the registers on the DT3362 series boards. The term "Base" refers to the device base address (see Section 3.2).

TABLE 5-1: DT3362 SERIES REGISTER ASSIGNMENTS

REGISTER	ADDRESS	ACCESS
A/D Control Status (ADCSR)	Base + 0	R/W: Byte
A/D Data Buffer (ADBUF)	Base + 02	Read: Word
Channel-list Programming (CLPR)	Base + 04	R/W: Byte
DMA Control Status (DMACSR)	Base + 06	R/W: Byte
DMA Word Count (DMAWCR)	Base + 10	R/W: Byte
DMA Address Counter DMACAR	Base + 12	R/W: Byte
Reserved	Base + 14	
Reserved	Base + 16	

NOTE:
1. R/W is Read/Write.

5.2.1 A/D CONTROL AND STATUS REGISTER

Address: Base + 0
Access: Read/Write

The A/D Control and Status Register (ADCSR) is a byte operable read/write register which controls and monitors all activity associated with the A/D converter subsystem. The ADCSR is shown in Figure 5-1. The functions of the bits are described next.

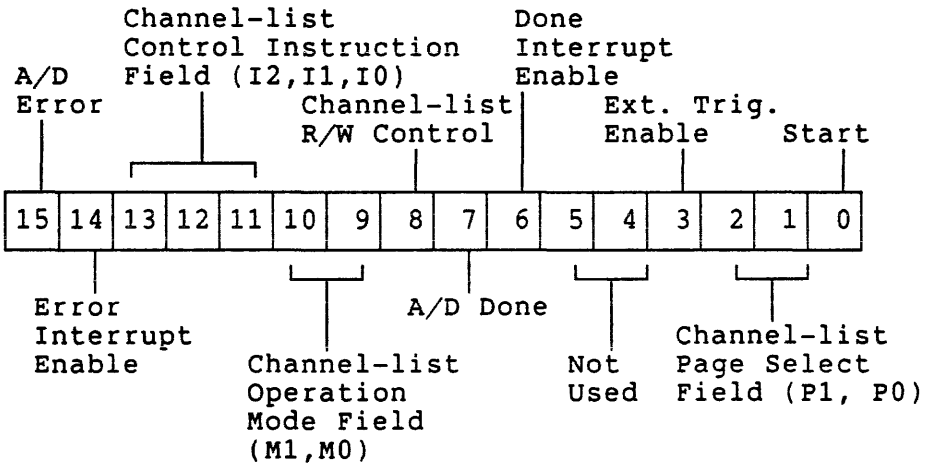


FIGURE 5-1: ADCSR BIT FUNCTIONS

BIT FUNCTIONS

BIT 15 - A/D Error, Read/Write to zero
A 1 indicates the occurrence of one of the following error conditions:

1. An attempt to start the converter with an external trigger while the multiplexer is settling (trigger overrun error).

2. Any attempt to start the converter when the data pipeline is full (data overrun error).

3. End of the second conversion before the data from the previous conversion is read (overrun error).

This bit is cleared at initialization or by a write to zero

BIT 14 - Error Interrupt Enable, Read/Write
A 1 allows the setting of the error bit (bit 15) to generate an interrupt request to the CPU. This bit is controlled by the program. It is cleared at initialization.

BITS 13-11 - Channel-List Control, Read/Write

This 3-bit instruction field is used to program the pointer logic that is used to access the channel-list RAM (see Section 5.3). It is valid only when mode 01 operation (Program Enable Pointer Logic) is selected in the mode field as described further below (bits 10, 9). In mode 01 operation, the 3-bit instruction field has the functions listed in Table 5-2. These three bits allow the user to program the channel-list address pointer. Actual data transfers are done through the high byte of the Base + 04 word location. An overview of the steps required to program the pointer logic is given in Chapter 8.

TABLE 5-2: CHANNEL-LIST CONTROL INSTRUCTION FIELD

ADCSR 13 I2	ADCSR 12 I1	ADCSR 11 I0	INSTRUCTION SELECTED
0	0	0	Write Control Register
0	0	1	Read Control Register
0	1	0	Read Final Address Pointer
0	1	1	Read Start Address Pointer
1	0	0	Reinitialize Counters
1	0	1	Load Current Address Pointer
1	1	0	Load Final Address Pointer
1	1	1	Not Used

BITS 10,9 - Channel-List Operation Mode, Read/Write

This 2-bit field specifies the particular operation required of the channel-list access counter logic. The four selectable operations are given below.

TABLE 5-3: CHANNEL-LIST OPERATION MODE FIELD

ADCSR 10 M1	ADCSR 9 M0	INSTRUCTION SELECTED
0	0	Enable Counters
0	1	Program Enable Counter Logic
1	0	Program Enable RAM File
1	1	Preload Multiplexer

BIT 8 - Channel-list Read/Write Control, Read/Write

This bit specifies the type of programming operation to be performed during either mode 1 or mode 2 of the channel-list operation as defined by bits 10 and 9. A zero indicates a read operation while a one indicates a write operation. This bit is controlled by the program.

BIT 7 - A/D Done, Read Only

When set indicates the completion of an A/D conversion. It is cleared when the A/D data buffer is read. This bit is initialized to zero.

BIT 6 - Done Interrupt Enable, Read/Write

A 1 allows the setting of the A/D Done bit (ADCSR bit 7) or the DMA Done bit (DMACSR bit 7) to generate an interrupt request to the Q-bus CPU. This bit is controlled by the program. It is initialized to zero.

BITS 5,4 - Not Used

These bits are not used and read back as zero.

BIT 3 - External Trigger Enable, Read/Write

A 1 enables the TTL compatible signal applied to the External Trigger input line to initiate device activity (single conversion for non-DMA; single conversion, List Scan, or Burst for DMA. See Chapter 6) on a high to low transition of the signal. When this bit is set, the Start bit (bit 0) is disabled from initiating board activity. This bit is controlled by the program. It is initialized to zero.

BITS 2,1 - Channel-list Page Select, Read/Write

These two bits allow the user to access statically one of four pages in the channel-list RAM. The user can prestore four unique sampling schemes in the list, each up to 256 entries long. At run time any one of the four sampling schemes may be selected. Table 5-4 summarizes the usage of these bits.

TABLE 5-4: CHANNEL-LIST PAGE SELECT

ADCSR 2 P1	ADCSR 1 P0	INSTRUCTION SELECTED
0	0	Select Page 0
0	1	Select Page 1
1	0	Select Page 2
1	1	Select Page 3

BIT 0 - Start, Read/Write

When set to 1, this bit initiates an A/D conversion in either DMA or non-DMA operating modes. This software trigger should be used only when the external trigger input is disabled. This bit is cleared automatically by the hardware. If external trigger operation is enabled then setting this bit will not initiate device activity and the bit will read back as 1. When this happens, you must write this bit back to a zero before clearing the external trigger enable bit. Failure to use the bit in this specified way could cause an extraneous A/D conversion to be made. This bit is initialized to zero.

5.2.2 A/D DATA BUFFER REGISTER

Address: Base + 02
Access: Read Only

The A/D Data Buffer Register (ADBUF) is a word operable read only register used to read A/D data for non-DMA operating modes. This 16-bit register is the second stage in the data pipeline and contains the next converted value to be read. Whenever this register is updated with new data from the A/D converter, the A/D Done bit (ADCSR bit 7) sets. A read of this register causes the A/D Done bit to be cleared indicating that the data has been transferred out by the CPU.

5.2.3 CHANNEL-LIST PROGRAMMING REGISTER

Address: Base + 04
Access: Read/Write

The Channel-list Programming Register (CLPR) is an indirect sub-data bus used for all data I/O operations to the multiplexer channel-list control logic. The register is divided into two 8-bit sub-busses used for I/O data to different portions of the channel-list control logic. Figure 5-2 summarizes the use of the CLPR for channel-list programming.

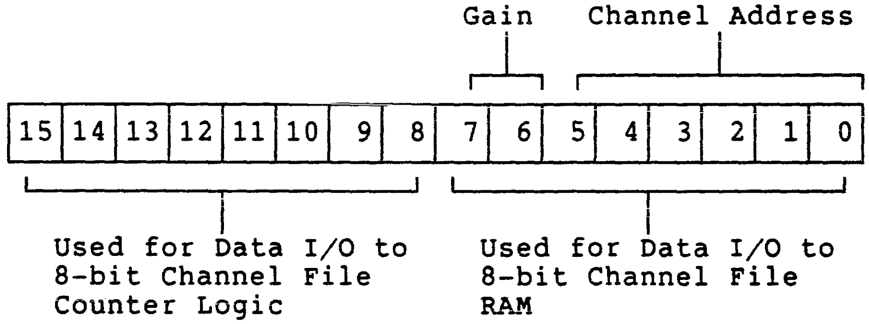

Gain
Channel Address
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
Used for Data I/O to 8-bit Channel File Counter Logic
Used for Data I/O to 8-bit Channel File RAM

FIGURE 5-2: CLPR BIT FUNCTIONS

This register is used in conjunction with the upper byte of the ADCSR which contains the channel-list operation control bits. The use of this register for different programming operations is summarized in Table 5-5.

TABLE 5-5: MODE SELECTION USING CLPR

MODE SELECTED	OPERATION REQUIRED	CLPR HIGH BYTE	CLPR LOW BYTE
Counter Enable	Active A/D operations	Not used	Not used
Channel-list Addressing Logic Programming	Write to a specified register	Data to be written to selected register	Not used
Channel-list Addressing Logic Programming	Read a selected register	Data read back from selected register	Not used
RAM Channel-list Programming	Write mux addresses sequentially into RAM	Not used	Data to be written to RAM file
Channel-list Programming	Read contents of RAM file for verification	Address of channel being read from RAM	Contents of RAM file location being read

5.2.4 DMA CONTROL AND STATUS REGISTER

Address: Base + 06
Access: Read/Write

All DMA related board activity on the DT3362 series is set up and controlled by the software via the DMA Control and Status Register (DMACSR). The DT3362 series can be used to perform DMA transactions either over the Q-bus or to dual-ported memory. The DMA data path to be used is selected through the DMACSR. It should be noted that the two DMA operating modes are mutually exclusive and both cannot be enabled together. If the software attempts to enable both DMA options simultaneously, the hardware will default to Q-bus DMA. The bits of the DMACSR are noted in Figure 5-3. A detailed discussion of the function of each bit is given next.

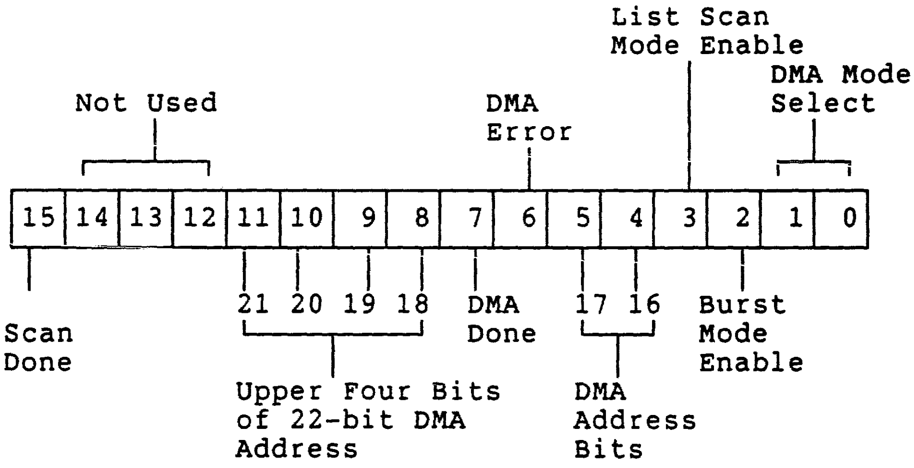


FIGURE 5-3: DMACSR BIT FUNCTIONS

BIT 15 - Scan Done, Read only
This bit sets each time the channel-list completes a scan. The bit resets automatically when a channel-list initialization occurs from either the internal circuitry or from a reinitialization command.

BITS 14-12 - Not used
These bits of the DMACSR are not used and are undriven on readback. Therefore, the state of these bits is indeterminate during read operations.

BITS 11-8 - 22-bit Address, Read/Write

These bits are used in conjunction with bits 5 and 4 (see the following discussion) and the sixteen bits of the DMA Current Address Register to generate a 22-bit DMA address. Bit 11 corresponds to address bit 21, bit 10 to address bit 20, bit 9 to address bit 19, and bit 8 to address bit 18. These bits are ignored if the board is jumpered for 18-bit DMA address range (jumper W59 out, W60 in). The bits are cleared on initialization. If DMA no-wrap mode is selected (jumper W58 is installed) these bits increment whenever an overflow occurs out of address bit 17.

BIT 7 - DMA Done, Read/Write

A 1 indicates the end of the current DMA operation. This bit can be used by the software as required, or can be made to interrupt via the Done interrupt logic. For this interrupt to occur the Done interrupt enable bit of the ADCSR must also be set. Note that the base interrupt vector associated with this interface is shared by both the A/D Done bit and the DMA Done bit. The hardware internally arbitrates the use of this interrupt according to the following criteria:

1. Interrupt to be used by A/D Done when interface is not operating in any of the DMA modes.

2. Interrupt to be used by the DMA Done when interface is operating in one of the DMA modes. The A/D Done line is inhibited from interrupting during DMA operation.

The DMA logic will end operation when a block of A/D data is transferred to memory as specified. Any memory error during DMA transfers aborts the DMA operation. If during the DMA operation, a memory error of any sort occurs, then the DMA operation will be aborted. This error condition sets the DMA Done bit in addition to the DMA Error bit. This bit can be written to zero by the program. It is also cleared on initialization.

BIT 6 - DMA Error, Read/Write to zero only

A 1 indicates that one of the two following types of memory errors has occurred:

1. Bus Time Out error when operating over the Q-bus. This error indicates that non-existent memory has been accessed over the Q-bus.

2. Access error to the dual-ported memory board. This error occurs when the A/D board is enabled to transfer data into the dual-port memory board, while the memory board has not been enabled.

Setting this bit during a DMA transfer aborts the remainder of the transfer and sets the DMA Done bit. This bit can only be written to zero by the program. It is also cleared on initialization.

BITS 5,4 - Q-bus Extended Address, Read/Write

This 2-bit field forms bits 16 and 17 of the address range on all versions of the DT3362 series boards. The field is dynamic in nature. If the no-wrap memory access mode is selected (jumper W58 in), the value of the field increments by one on each overflow of the DMA Current Address Register, and access starts on the next 64 Kbyte memory segment. If the wrap mode of memory access is selected, (jumper W58 out), the field value remains static and the DMA address wraps back to the beginning of the current 64 Kbyte segment following an overflow of the DMA Current Address Register. These bits have no effect on the LSI-11 and LSI-11/2 systems, but provide 18-bit memory addressing capability in LSI-11/23 systems where the 18-bit memory addressing feature has been incorporated. Bit 4 corresponds to address bit 16; bit 5 to address bit 17. These bits controlled by the program. They are cleared on initialization.

BIT 3 - List Scan Mode Enable, Read/Write

A 1 allows the board to operate in List Scan mode during DMA operation (see Section 6.4 for a description of the List Scan mode). This bit and bit 2 (Burst mode enable) of the DMACSR must not be high together. If they are both set high, one disables the other, resulting in the deactivation of the operation in both modes. This bit is controlled by the program. It is cleared on initialization.

BIT 2 Burst Mode Enable, Read/Write

When set this bit allows the board to operate in Burst mode during DMA operation (see Section 6.4 for a description of the List Scan mode). This bit and bit 3 (List Scan Mode Enable) of the DMACSR must not be high together. If they are both set high, one disables the other, resulting in the deactivation of the operation in both modes. This bit is controlled by the program. It is cleared on initialization.

BITS (1, 0) DMA Mode Select, Read/Write

This 2-bit field specifies the type of DMA activity required. In essence these bits enable or disable the DMA logic and select whether the DMA is to occur over the Q-bus or the External Port. Table 5-6 summarizes the use of these bits.

TABLE 5-6: DMA MODE SELECTION

DMACSR 1	DMACSR 0	DMA MODE DESCRIPTION
0	0	NOP DMA operation disabled. Board to operate in programmed I/O or interrupt mode.
0	1	Enable DMA operation to Q-bus.
1	0	Enable DMA operations to Dual-ported memory.
1	1	Enable DMA operation to Q-bus.

5.2.5 DMA WORD COUNT REGISTER

Address: Base + 10
Access: Read/Write

The DMA Word Count Register (DMAWCR) is a 16-bit read/write register used only when a DMA transfer via the Q-bus is to be performed. The register is loaded with the two's complement of the number of data conversions, and is incremented after each word transfer. The DMA operation is ended and a DMA Done interrupt issued when this register increments to zero.

5.2.6 DMA CURRENT ADDRESS REGISTER

Address: Base + 12
Access: Read/Write

The DMA Current Address Register (DMACAR) is a 16-bit read/write register that is used only when a DMA transfer via the Q-bus is to be performed. In this operating mode this register is loaded with the starting word address of the reserved memory buffer into which the data is to be transferred and is incremented after each word transfer, thus causing data to be stored in memory sequentially and in ascending order.

5.3 MULTIPLEXER CHANNEL-LIST

The multiplexer channel-list is an on-board 1024x8 RAM which is divided into four 256-byte pages. The processor uses each page to store the channel numbers in the sequence in which they are to be sampled. The processor accesses all four pages of the channel-list via two static page select bits in the ADCSR. Therefore any sampling scheme may be up to 256 bytes long and the processor can preload four different sampling schemes into the list.

Access to the channel-list from both the processor and the conversion logic is controlled via the 8-bit incrementable Current Address Pointer which has an associated 8-bit Start Address Register and an 8-bit Final Address Register. (These two registers are selected via the channel-list instruction field in the ADCSR and are accessed via the Channel-list Programming Register as explained earlier in this chapter.) The logic is designed such that the CPU can access the channel-list only when the latter is inactive (when no conversions are in progress). When the device is active, the CPU is locked out from channel-list access.

On all DT3362- prefixed models and DT3377, the 8-bit pointer scheme for accessing the channel-list allows the processor to set up the board to scan only as much of the channel-list, within a 256 byte block, as is required during a certain operation. The channel access pointer automatically increments after each conversion until it matches the contents of the Final Address Register. At that time it resets back to the value held in the Start Address Register.

On DT3368-4SE and DT3368-12SE, the multiplexer channel-list operates differently. Here the channel-list can have any sampling sequence but must always start with Channel 0. No channel address can be repeated in the list. The channel address pointer increments automatically after each A/D conversion until it matches the contents of the Final Address Register. At the end of the current scan, the multiplexer returns to Channel 0, all channels are placed back in the sample mode, and the operation is terminated until a new conversion trigger is issued.

On boards equipped with a programmable gain, each byte stored in the channel-list contains a 2-bit field which selects the gain to be applied on the input.

5.4 ADDRESS POINTERS

The pointer logic of the DT3362 series board contains three pointers which indicate channel addresses in a Channel-list Scan operation. They are the Start Address Pointer, Current Address Pointer, and Final Address Pointer. The three pointers are selected via the Channel-list Control instruction field in the ADCSR (bits 11, 12, and 13).

The Start Address Pointer holds the starting channel address of the current Channel-list scan. The Current Address Pointer indicates the address to be selected from the Channel-list for the current A/D operation. The Final Address Pointer indicates the last channel to be selected in the current A/D conversion cycle. The values stored in the Starting Address Pointer and the Final Address Pointer remain constant throughout the scan operation, while the content of the Current Address Pointer increments following completion of each A/D conversion. When the content of the Current Address Pointer equals the value stored in the Start Address Pointer, the Current Address Pointer automatically resets (presets) to the value stored in the Start Address Pointer. The Start Address Pointer and the Current Address Pointer are each 8-bit wide. They are loaded with a single 16-bit word.

5.5 EXTERNAL PORT

As Figure 5-4 illustrates, the DT3362 series use the Q-bus (system bus) for data transfers to system memory. This type of conventional architecture places an additional burden on the system since the data conversion subsystem must share the Q-bus with other peripherals requesting service from the CPU. Added traffic on the Q-bus creates congestion and ultimately degrades both the performance of the system with respect to other peripherals and throughput of the A/D conversions. This situation may be permissible if high demands are not made of the system. For many applications requiring high throughputs, however, latencies due to high traffic on the Q-bus could be intolerable.

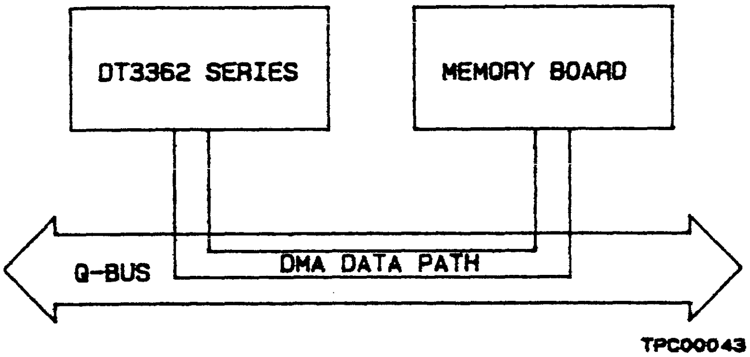


FIGURE 5-4: DATA PATH TAKEN BY CONVENTIONAL DMA TRANSFERS

The DT3362 series board features an External Port which is designed to be used for data transfers external to the system bus. A memory device used in conjunction with the External Port appreciably alleviates congestion on the Q-bus. The DT3362 series board transfers conversion data to the memory via the External Port which operates independently of the Q-bus. The data is then transferred to a mass storage media via the Q-bus.

In this type of architecture, data transfers occur over the External Port without interrupting other activity on the Q-bus. The External Port of the DT3362 series board can be linked to a compatible external device such as the DT3369 dual-port memory board or the SKY 320 signal processor. Architecture including the DT3369 dual-port memory device, for instance, dramatically enhances the throughput of the DT3362 series board as well as the performance of other subsystems on the Q-bus. Figure 5-5 shows the External Port in relation to the Q-bus. The following two subsections describe the two devices, mentioned above, that can be used with the DT3362 series board for high performance applications.

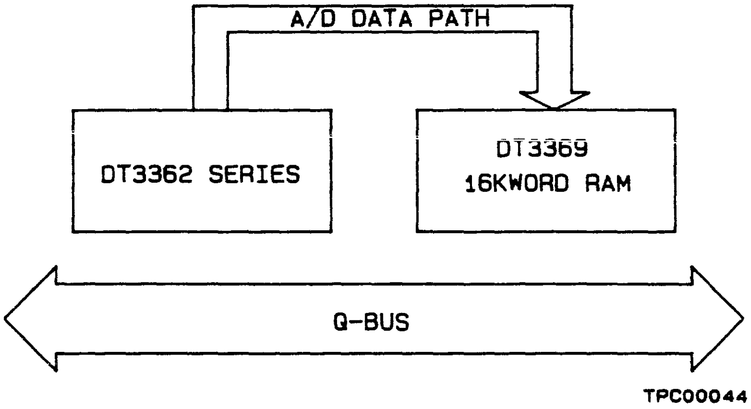


FIGURE 5-5: DATA PATH TAKEN WITH DUAL-PORTED MEMORY STRUCTURE

5.5.1 DT3369 DUAL-PORT MEMORY

The DT3369 is a dual-port RAM board. When combined with the DT3362 series board, it changes drastically the architecture of the data acquisition system. The dual-port memory board appears to the data acquisition system as a standard memory subsystem, but it is totally independent of the Q-bus. One port of the DT3369 communicates with the External Port on the DT3362 series, the other with a mass storage peripheral over the Q-bus. In case of a simultaneous access requests to both ports of the external dual port memory, the DT3369 on-board controller resolves the conflict so that each side is transparent to the other. The operation of the dual port external memory is explained in Section 5.5.

5.5.2 SKY 320 SIGNAL PROCESSOR

The SKY 320 signal processor operates over the External Port in the same manner as the DT3369 dual port memory board discussed in the previous subsection. Data is acquired by the A/D subsystem and passed through directly to the signal processor without flowing through the Q-bus or requiring the intervention of the host processor. Thus the addition of the SKY 320 in the I/O space of the system processor does not increase the activity on the Q-bus, and, therefore, does not add to the bus latency of the system.

5.6 DMA TRANSFERS

The DT3362 series boards contain circuitry which permits direct data transfers between the board and the system memory without the intervention of the CPU. This type of data transfer is termed Direct Memory Access (DMA). If the DT3362 series board is operating by itself, then all DMA transfers are performed over the Q-bus. However, if the DT3362 series board is used in combination with the DT3369 external memory board, then DMA transfers can take place over the External Port as well as the Q-bus. This section explains how data transfer operations differ on these two busses.

5.6.1 DMA TRANSFERS VIA Q-BUS

Data transfer via the Q-bus is the conventional DMA data path for Q-bus peripherals and allows the DT3362 series boards to be used by themselves as high speed data acquisition peripherals. Since the Q-bus is the transmission medium for all data transfers, it must arbitrate the transfer of data between the A/D board and memory, the transfer of instructions from memory to the CPU, and the transfer of blocks of data between memory and the storage peripherals. Time spent in arbitration can significantly reduce the effective bandwidth of the processor bus.

Furthermore, use of the Q-bus DMA means that during data output all other bus related tasks and the CPU must be temporarily halted. As the throughput requirements increase, the amount of time the A/D peripheral spends on the bus increases and therefore the bus free time decreases. The limit on throughput is achieved when the bus free time goes to zero. At that point, the bus is being used solely to transfer data between the A/D board and memory. No other activity can be performed. In many applications requiring large amounts of high speed data to be collected, stored on disk, and played back, this architecture can be limiting, as explained above. The DT3369 dual-port external memory device relieves activity on the Q-bus and maintains high performance of the system.

5.6.2 DMA TRANSFERS VIA EXTERNAL PORT

The DT3362 series board contains circuitry that allows it to link directly with one side (port) of the DT3369 dual-port memory, creating a data bus which operates with complete independence from the Q-bus (processor bus). On the other side, the dual-port memory connects to the Q-bus and appears to the processor as a standard memory subsystem.

Moving data between the board and memory over the External Port makes data acquisition rates independent of the Q-bus arbitration latency or performance. With the standard DMA approach, data moves twice over the Q-bus: first, from A/D to memory, then from memory to disk. With the dual port architecture, the Q-bus is needed only to move data from memory to disk. Thus an equivalent number of Q-bus transactions are handled by the dual-port system with half the usual Q-bus overhead.

The dual port memory subsystem is further enhanced by features available on the DT3369 memory board. The on-board address controller can direct the system to alternately load data into two separate sub-buffers on the DT3369 board. The filling of each sub-buffer can interrupt the processor, which then initiates a DMA transfer via Q-bus from memory to disk. While one buffer is being emptied via Q-bus, the A/D can continue to fill the other buffer via the External Port. This process can be continued indefinitely, achieving gap-free acquisition, limited only by the transfer rate and size of the disk.

CHAPTER 6

OPERATING PRINCIPLES

6.1 INTRODUCTION

The data acquisition and conversion operations on the DT3362 series boards go through three processes: 1) A/D conversion trigger; 2) Data conversion; and 3) Data transfer. This chapter explains the various modes of operation which the user can define for the board by appropriately selecting between alternatives available for each of these three processes.

6.2 A/D CONVERSION TRIGGER

The DT3362 series boards initiate A/D conversions by a trigger issued either via software (software trigger) or an external source (external trigger). A software trigger is enabled when bit 0 of the ADCSR is set while bit 3 of the ADCSR is clear. Data conversion starts as soon as the ADCSR is written to. An external trigger is enabled by setting bit 3. The board then waits for an external trigger source to initiate data conversion. Once enabled, any TTL-compatible pulse train can be tied to the external trigger input line to start conversions.

On all models a single trigger activates A/D conversions in one of three modes: Single-channel, List Scan, or Burst. In the Single-channel mode a trigger is required for each conversion. In the List Scan Mode the channels selected in the multiplexer channel-list (see Section 5.3) are scanned and converted with each trigger. The process ends when the last channel in the list is converted. In the Burst mode, the trigger sets off a series of conversions. The operation ends when the number of conversions equals the number specified in the DMA Word Count Register. On all models except DT3368-4SE and DT3368-12SE, the A/D module can sample any channel in any desired sequence.

On models DT3368-4SE and DT3368-12SE, which feature a simultaneous sample and hold A/D converter module, the sampling scheme requires that Channel 0 be sampled first, followed by at least one other channel. No channel can be sampled twice without Channel 0 being sampled in between. The sequence of events for the simultaneous sample and hold is given below.

When Channel 0 is triggered:

1. All sample and hold circuits freeze and switch to the hold mode.

2. The channel-list is scanned at 100kHz, converting each analog input to a digital value.

3. The multiplexer switches back to zero, the first element in the list.

4. Switching to Channel 0 from some other channel causes all sample and hold circuits to be unfrozen and go back to sample (track) mode.

The next occurrence of a trigger will repeat this cycle.

6.3 DATA TRANSFER

There are three modes of data transfer in a data acquisition system based on the DT3362 series board and the DT3369 dual-port memory board: Programmed I/O, DMA transfer over the Q-bus, and DMA transfer over the External Port. The data transfer mode is selected by loading the appropriate value into the DMA mode select field in the DMACSR (bits 1 and 0, see Table 5-6).

In the Programmed I/O mode, the CPU transfers each word of data to the host memory. The CPU can poll the DT3362 series board to determine when each conversion is complete, or, alternatively, the board can be made to generate an interrupt to the processor when a conversion is complete. The CPU then reads the converted data and stores it in the memory.

In the Q-bus DMA mode, the starting address is loaded into the DMACAR and the DMACSR. The two's complement of the number of words to be transferred is loaded into the DMA Word Count Register (DMAWCR). Each time a data word is transferred across the bus, the DMAWCR and the DMACAR increment by one.When the DMACAR reaches 0, the board sets the DMA Done bit and either generates an interrupt or waits to be polled.

In the External Port DMA mode the number of words to be transferred and the destination address are set by the device at the other end of the External Port. Data is transferred only over the External Port. Consult the DT3369 Dual-Port memory user manual for additional information on data transfer over the External Port.

6.4 DATA CONVERSION

Three modes of data conversion are available: Single Conversion per trigger, List Scan per trigger, and Burst Conversion per trigger.

The Single Conversion per trigger is set by clearing bits 2 and 3 in the DMACSR. In this mode, the board collects or writes one data point for each trigger (software or external) issued. After the data has been collected or output, the board waits for another trigger to acquire another data point. The single conversion triggers operate in both the Programmed I/O and DMA modes of data transfer.

The List Scan mode is selected by setting bit 3 in the DMACSR while bit 2 is clear. Once the process has been initiated, the board continues automatically to perform conversions until one scan has been done. A scan is defined as the channel-list Current Address Pointer reaching the value stored in the Final Address Register. When a scan is complete, the Current Address Pointer resets to the value held in the Start Address Register and the board waits for another software start or external trigger. This process continues until the total number of conversions specified have been taken. The List Scan mode is only available with the DMA modes of data transfer.

The Burst Conversion mode is selected by setting bit 2 in the DMACSR while bit 3 is clear. In this mode the board proceeds to convert another data point as soon as the previous conversion is complete. Once the process has been initiated, the channel-list is scanned repeatedly until the number of words specified have been collected or written. The Burst Conversion mode is available only with the DMA modes of data transfer.

CHAPTER 7

CHANNEL-LIST PROGRAMMING

7.1 INTRODUCTION

The DT3362 series offers the user a sophisticated and highly versatile scheme for defining the channels to be sampled. This feature, called the multiplexer channel-list, allows you to program into the interface the channel numbers in the order in which they are to be sampled. Furthermore, the you can at any time request that a smaller portion of the channel-list be scanned.

On all models, the channel-list is organized as four pages of 256 bytes per page. Page selection is done by a 2-bit code in the ADCSR. Within a given page you can access the 256 bytes through the 8-bit pointer logic. Programming the channel-list logic consists of programming the pointer logic to operate within a specified starting and ending value, then loading the channel number information into the on-board RAM.

The processor has read/write access to the RAM so that it can load channel numbers to effect any desired sampling sequence For models featuring the simultaneous sample and hold module (DT3368-4SE and DT3368-12SE), however, the channel number sequence has two restraints: 1) the list must always start with Channel 0; 2) no channel can be converted more than once at the end of the sample phase (selection of Channel 0). If a channel is to be converted twice, then the module should be placed in the sample mode after the first conversion by selecting Channel 0. This process acquires new analog data at all the channels. A new conversion can now be performed on the selected channel.

In the following subsections detailed descriptions of the various programming steps required are presented. Actual programming of the interfaces then consists of assembling these operations in the required sequence.

NOTE

> In programming examples, it is the convention in this manual to underline the code which should be entered by the user.

7.2 PROGRAMMING PROTOCOLS

The following sections present the various protocols and sequences needed to program the on-board multiplexer channel-list and pointer logic.

7.2.1 INITIALIZE POINTER LOGIC

Following a reset or power-up, the pointer logic for the channel-list RAM has to be initialized before the board is used. This initialization is performed as follows:

1. Write 003 into the high byte of the ADCSR. This selects the write control register function in mode 1 with write selected.

2. Write the initialization code 002 into the pointer field (high byte) of the CLPR (Base + 04).

A typical instruction sequence for initialization would be:

```
.
.
MOVB #3, @#ADCSR + 1
MOVB #2, @#CLPR + 1
.
.
.
```

If the programmer wishes to read back the initialization code from the interface for diagnostic and testing purposes, the following sequence is required:

1. Write 012 into the high byte of the ADCSR.

2. Read the high byte of the CLPR to get the initialization code from the interface.

Programming sequence:

```
.
.
MOVB #12, @#ADCSR + 1
MOVB @#CLPR + 1, R0
.
.
.
```

7.2.2 WRITE START ADDRESS REGISTER/CURRENT ADDRESS POINTER

The 8-bit Start Address Pointer is loaded in parallel with the 8-bit Current Address Pointer when the program explicitly writes to the Current Address Pointer. After initial loading the program can at any time reset the pointer to the value held in the Start Address Pointer by issuing a reinitialization command to the pointer logic. The pointer will also be preset to the value in the Start Address Pointer when the content of the pointer become equal to the content of the Final Address Pointer during an active data acquisition cycle. The sequence required to write a value into the Start Address Pointer/Current Address Pointer is:

1. Write 053 into the high byte of the ADCSR.

2. Write the 8-bit value into the high byte of the CLPR.

Programming sequence:

```
.
.
MOVB #53, @#ADCSR + 1
MOVB #7, @#CLPR + 1
.
.
```

This will initialize the pointer and preset to 7.

7.2.3 READ CURRENT ADDRESS POINTER

If during the data acquisition cycle the program must determine the value of the pointer or must read, for a diagnostic test, the value of the pointer after having written to it, the following sequence is required:

1. Write 032 into high byte of ADCSR

2. Read contents of address pointer at high byte of CLPR

Programming sequence:

```
.
.
MOVB #32, @#ADCSR + 1
MOVB @#CLPR + 1, R0
.
.
```

7.2.4 WRITING FINAL ADDRESS POINTER

The Final Address Pointer specifies the value at which the Current Address Pointer should set itself back to the (initial) value held in the Start Address Register. The sequence required to write a value into the Final Address Register is:

1. Write 063 into the high byte of the ADCSR.

2. Write the desired value into the high byte of the CLPR.

Programming sequence:

```
.
.
.
MOVB #63, @#ADCSR + 1
MOVB #20, @#CLPR + 1
.
.
.
```

This example would load the number 20 into the final address register.

7.2.5 READING THE FINAL ADDRESS POINTER

For test, diagnostic, and verification purposes, the final address register may have to be used. This read operation can be performed with the following programming sequence:

1. Write 022 into the high byte of the ADCSR.

2. Read the value of the final address register at the high byte of the CLPR.

Programming sequence:

```
.
.
MOVB #22, @#ADCSR + 1
MOVB @#CLPR + 1, R0

.
.
.
```

7.2.6 RESETTING THE POINTER

At certain times during the programming of the interface it may be necessary to reset the Current Address Pointer to the value held in the Start Address Register. This can be accomplished easily using the pointer reinitialization command provided. To perform this function the following sequence is required:

1. Write 043 to the high byte of the ADCSR

2. Perform a write operation to the high byte of the CLPR. Since data written is not used, it can be any value.

Programming sequence:

```
        .
        .
        .
        MOVB #43, @#ADCSR + 1
        MOVB #0, @#CLPR + 1
        .
        .
        .
```

7.3 ENTERING CHANNEL INFORMATION INTO THE RAM

The on-board channel-list is 8-bits wide to allow the user to select any one of 64 channels (0 to 63) which requires a 6-bit field, and associate with that channel a code for the programmable gain amplifier (PGH = 1,2,4,8) which requires a 2-bit field. An 8-bit entry in the channel-list is organized as indicated in Figure 7-1:

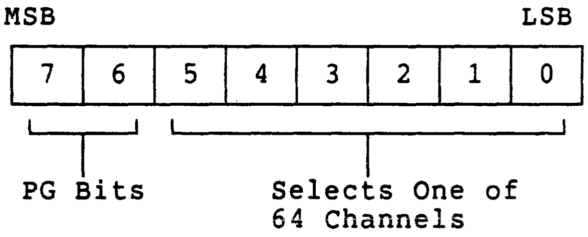


FIGURE 7-1: FORMAT OF CHANNEL-LIST ENTRIES

The bits select one of four possible gains. If a PG option is not installed, these bits are ignored. Gain selection is done according to Table 7-1.

TABLE 7-1: CHANNEL-LIST GAIN SELECTION

PG BIT 7	PG BIT 6	PGH GAIN
0	0	1
0	1	2
1	0	4
1	1	8

Each page of the channel-list is independent of the other three. Each has to be individually programmed by the user to the required sampling scheme. During actual conversion operations, only one of the four pages as selected by the two-bit field of the ADCSR is accessed through the pointer logic. The user can program and read back for verification the contents of the channel-list RAM by following the programming sequence given in the ensuing two sections.

7.3.1 WRITING INTO THE CHANNEL-LIST RAM

Following initialization, you must first select which one of four pages in the channel-list is to be operated on by writing a two bit page select code into bits 2 and 1 of the ADCSR. Then the channel-list pointer logic must be set to the required starting address and be told of the desired ending address. The pointer logic points to the first location in RAM that is to be loaded. The RAM accepts data for entry in the following manner:

1. Write a 005 into the high byte of the ADCSR.

2. Write the sequence of channel numbers into the low byte of the CLPR.

Note that the pointer will be incremented after each write operation to the low byte of the CLPR. Thus sequential writes to the low byte of the CLPR will place data sequentially in the channel-list. This incrementing operation will proceed until the value of the pointer becomes equal to the value held in the final address register. A write operation performed after this condition will set the pointer back to its initial value and the channel-list buffer will be overwritten.

Programming sequence:

```
.
.
.
MOV  #2400, @#ADCSR
MOVB #1, @#CLPR
MOVB #3, @#CLPR
MOVB #5, @#CLPR
MOVB #7, @#CLPR
.
.
```

This example selects page 0 of the RAM and forces a list programming operation. Then the odd channel numbers are stored sequentially into the channel-list.

7.3.2 READING THE CHANNEL-LIST RAM

For diagnostic testing, and verification purposes the contents of the channel-list RAM may have to be read. To perform read operations the program must first set the pointer logic to the desired starting address and specify the required final address. Then the following sequence is required:

1. Write 004 into the high byte of the ADCSR.

2. Read the channel-list entries successively from the CLPR low byte. Note that when the CLPR is read in this mode, for diagnostic purposes, the high byte of the register will contain the address of the channel-list entry that is being read while the low byte will contain the contents. The pointer will be incremented after each read operation. Hence successive reads will allow the program to read successive entries of the RAM. The pointer logic is active during this operation. Therefore, when the pointer increments to the final address register, it will be reset back to the starting address.

Programming sequence:

```
.
.
.
MOV #2000, @#ADCSR
MOV @#CLPR, R0
MOV @#CLPR, R1
MOV @#CLPR, R2
.
.
```

This example selects page 0 for readback purposes and then takes three successive entries from the list and places them in registers R0, R1, and R2.

7.3.3 MULTIPLEXER PRELOAD AND LOGIC ENABLE

After the pointer logic has been programmed and the Channel-list RAM has been loaded with appropriate data the program must set the pointer logic for an active cycle. Part of this activation involves issuing a load command to the actual multiplexer and then enabling the logic for A/D conversion operations. The following sequence is required for this:

1. Write 006 to the high byte of the ADCSR.

2. Perform a write operation to the low byte of the CLPR. Note that the data written is not critical as only the write operation is required.

3. Write 000 to the high byte of the ADCSR

Programming sequence:

```
        .
        .
        MOVB #6, @#ADCSR + 1
        MOVB #0, @#CLPR
        MOVB #0, @#ADCSR + 1
```

After this operation you must access only bits 14 and 15 of the high byte of the ADCSR to ensure that the channel-list is not disturbed.

CHAPTER 8

PROGRAMMING EXAMPLES

8.1 INTRODUCTION

This chapter presents five programs that describe how the DT3362 series board should be programmed for certain non-DMA operations or DMA transfers via the Q-bus. The programs are not intended to solve any particular application problem but have been written such that key programming issues are demonstrated. Programs 1 and 2 illustrate non-DMA operations, while programs 2, 3, and 5 illustrate DMA transfer operations. A brief summary precedes the listing of each program. All example programs have been written as subroutines so that you can test them by calling them from your main program. Note again that these example programs are given for illustration only and should not be used for diagnostics.

8.2 FADEX1 - EXAMPLE PROGRAM 1

In this example the DT3362 series board is used in the programmed I/O mode to perform a single conversion on Channel 5 with a gain of 4 every time it is called. The subroutine initializes the multiplexer channel-list and loads the appropriate channel and gain information. Following this, it initializes the A/D section and starts a programmed conversion. The program then loops waiting for the A/D Done status bit to set. When this occurs the A/D data is read back along with status information and a return of subroutine is done.

8.3 FADEX2 - EXAMPLE PROGRAM 2

In this example the DT3362 series board is set to perform externally clocked conversions on Channel 5 at a gain of 1, Channel 5 at a gain of 2, Channel 7 at a gain of 1, and Channel 2 at a gain of 1. Each external clock initiates one conversion. Four clock ticks complete one scan of the channel-list. The board interrupts the processor after every conversion, and the interrupt handler keeps track of the number of interrupts received. After 100 interrupts are received (25 scans), the interrupt handler disables the A/D from further conversions and write into the status word. Thus if a main program calls this subroutine after clearing the status, it can then test the status word for a non-zero value signifying the completion of an A/D conversion.

8.4 FADEX3 - EXAMPLE PROGRAM 3

In this example the DT3362 series board will be set to perform a burst of 100 conversions and place them in main memory via DMA over the Q-bus. The channel-list is set up to sample in the following sequence.

1) Channel 5 @ gain = 1
2) Channel 5 @ gain = 2
3) Channel 7 @ gain = 1
4) Channel 2 @ gain = 1

Thus each scan of the channel-list will be 4 conversions requiring the channel-list to be scanned 25 times to get the 100 conversions. The burst mode operation implies that the A/D is running at its maximum speed in that each conversion is initiated by the end of the previous conversion until the required number of conversions have been made.

Due to bus access requirements of a 250kHz A/D converter (DT5727) the program places the processor in the wait state to free up the bus. To avoid having an A/D overrun error, you must also make sure that no other DMA activity is happening during the burst A/D. This ensures that A/D conversion is complete when control is returned to the calling program.

8.5 FADEX4 - EXAMPLE PROGRAM 4

The example program is essentially the same as number three in that a total of 100 conversions are to be made and put into memory under DMA. The channels to be sampled are also identical. The only difference is that the rate of conversions is now controlled by the external clock input. Thus the time interval between conversions is equal to the period of the externally applied clock. Here a single conversion per clock tick is made, requiring 100 clock ticks to perform the necessary 100 conversions. An assumption has been made that the clock frequency should not exceed 100kHz to allow the DT3362 series adequate time to transfer data over the bus in the face of processor and other bus activity. The main program should call this subroutine after having cleared the PSTAT flag word. Then the main program can test when PSTAT becomes non-zero to determine that the DMA transfer has completed.

8.6 FADEX5 - EXAMPLE PROGRAM 5

This program is again essentially the same as the previous two in that 100 conversions with the same channel-list are to be made and put into main memory. The only difference is that the external trigger with List Scan mode is selected. In this mode each external clock tick will cause the DT3362 series to scan through the channel-list once. Thus each scan of the list will produce four conversions, requiring 25 external clock ticks to gather 100 data points. As the DT3362 series is bursting through the channel-list at its maximum speed, a processor wait is required to ensure that the A/D had full access to the Q-bus to avoid overrun errors. Here again, control returns to the calling program when the required number of data conversions have been taken.

```
        .ENABLE LC
        .TITLE FADEX1  Fast A/D (DT3362) Programming Example 1
;
        .SBTTL Functional description
;
;  Tested:   25 Mar 81
;
;  This routine is a programming example designed to operate the
;  Data Translation Fast A/D converter: DT3362 .
;
;  In this example the Converter is required to make a single
;  conversion in Programed I/O mode.  The channel and gain values
;  are specified in the data block below.  Here channel #5 is
;  selected with a gain of 4.
;
;
        .SBTTL  A/D data block, and control block definitions.
;
BASE   = 170400          ; Base address of Fast A/D converter.
;
ADCSR  = BASE            ; A/D Control and Status Register
ADBUF  = BASE+2          ; A/D Data Buffer Register
CFPR   = BASE+4          ; Channel File Programming Register.
DMACSR = BASE+6          ; DMA Control & Status Register.
DMAWCR = BASE+10         ; DMA Word Count Register. (not used)
DMACAR = BASE+12         ; DMA Current Address Register. (not used)
;
INVEC  = 400             ; Address of Interrupt vector. (not used)
;
CHAN:   005              ; Channel desired.
GAIN:   002              ; Gain specifier (02 is gain of four.)
DATA:   .BLKW   1        ; Space for A/D data.
STATS:  .BLKW   1        ; Space for A/D Completion status.
;
;
FADE1::                  ; This is the routines entry point.
;
        .SBTTL  Initialization (in case of powerup or reset)
;
;   This initialization code need only be executed once after
;   a powerup or hardware reset.
;
        MOV     #001400, @#ADCSR        ; select write to channel file
        MOVB    #002, @#CFPR+1          ; write required setup code.
;
;
        .SBTTL  Setup Channel & Gain Array
;
;  This section sets up the channel file array.  This array specifies
;  which multiplexer channels are to be selected and then converted.
;
;  First set up Preset pointers, these indicate which portion of the
;  channel array is to be used.  This example uses the first byte of
;  the first page of the channel file.
;
```

```
;
        MOVB    #053, @ #ADCSR + 1      ; select start addrs., mode 01
        MOVB    #000, @ #CFPR + 1       ; set initial pointer to zero.
;
        MOVB    #063, @ #ADCSR + 1      ; select final addrs., mode 01
        MOVB    #000, @ #CFPR + 1       ; set end pointer to zero.
;
;   Now to form the gain and channel byte.
;
        MOV     GAIN, RO                ; Get the gain information.
;
;   Must move Gain value over to bits 6-7
;
        SWAB    RO                      ; Move left 8 bits
        ASR     RO                      ; Move back (right) one bit
        ASR     RO                      ; Move back (right) one bit
;
;   Now ADD in channel value.  (ADD is Logical OR if no bits match)
;
        ADD     CHAN,RO                 ; ADD (OR) in channel select
;
;   First select channel array page zero and starting word 0
;
        MOV     #002400, @ #ADCSR       ; page 00, mode 10 and byte 0
;
;   Then write the channel and gain byte into the first location.
;
        MOVB    RO, @ #CFPR             ; gain and channel are in RO
;
;   Now Preload and Enable the Multiplexer logic.
;
        MOVB    #006, @ #ADCSR + 1      ; Preload Multiplexer, mode 11
        MOVB    #000, @ #CFPR           ; Give it a kick.
        MOVB    #000, @ #ADCSR + 1      ; Enable counters, mode 00
;
;   Now set DMACSR to indicate use of Programed I/O method.
;
        MOVB    #000, @ #DMACSR         ; Select Programmed I/O mode.
;
;   Now start converter by setting start bit in ADCSR.  Note that the
;   other control bits like channel page select and trigger ext must
;   be set correctly also.
;
        MOVB    #001, @ #ADCSR          ; Set start bit.
;
;
;   Now the converter is running.  This example will continue to test
;   the A/D Control & Status register looking for the completion bit.
;
RTEST:  TSTB    @ #ADCSR        ; If done bit set, byte is negative
        BPL     RTEST           ; If it hasn't changed loop around.
;
;   Program continues when done bit is set.  This should indicate that
;   data has been collected.
;
        MOV     @ #ADBUF, DATA  ; Get data from converter board.
        MOV     @ #ADCSR, STATS ; Now store DMA status.
;
        RTS     PC              ; return to calling routine
        .END
```

```
.ENABLE LC
.TITLE FADEX2  Fast A/D (DT3362) Programming Example 2
;
;       .SBTTL Functional description
;
;  Tested: 25 Mar 81
;
;  This routine is a programming example designed to operate the
;  Data Translation Fast A/D converter: DT3362 .
;
;  In this example the Converter is required to make one hundred
;  conversions in Interrupt driven mode.  These conversions will each
;  be initiated by the external clock.  The multiplexer channels
;  and their gain values are specified in the data block below.
;
;  In this example the channel array specifies that channel #5
;  will be read with a gain of 1, then again with a gain of 2,
;  then channel #7 will be read with a gain of 1, and finally
;  channel #2 will be read with a gain of one.
;
;  The interrupt service routine appears at the bottom
;  of this listing.
;
;
        .SBTTL  A/D data block, and control block definitions.
;
BASE   = 170400         ; Base address of Fast A/D converter.
;
ADCSR  = BASE           ; A/D Control and Status Register
ADBUF  = BASE+2         ; A/D Data Buffer Register
CFPR   = BASE+4         ; Channel File Programming
DMACSR = BASE+6         ; DMA Control & Status
DMAWCR = BASE+10        ; DMA Word Count Register. (not used)
DMACAR = BASE+12        ; DMA Current Address Register. (not used)
;
INVEC  = 400            ; Address of Interrupt vector.
;
;  Gain & channel selection array follows
;
GANDC:  .BYTE   005     ; Gain 1 (code 0), channel 5
        .BYTE   105     ; Gain 2 (code 1), channel 5
        .BYTE   007     ; Gain 1 (code 0), channel 7
        .BYTE   002     ; Gain 1 (code 0), channel 2
;
;  The Data descripters are reset each time the startup code is
;  executed.
;
PDATA:  DATA            ; Pointer to Data array.
DATA:   .BLKW   100     ; Data array.
;
DCNT:   100.            ; Data count. (number of conversions)
;
PSTAT:  STATS           ; Pointer to status word.
STATS:  .BLKW   2       ; Space for two status words.
;
FADE2::                 ; This is the routines entry point.
;
```

```
        .SBTTL  Initialization (in case of powerup or reset)
;
;    This initialization code need only be executed once after
;    a powerup or hardware reset.
;
        MOV     #001400, @#ADCSR+1      ; select write to channel file
        MOVB    #002, @#CFPR+1          ; write required setup code.
;
        .SBTTL  Setup the interrupt vector
;
;   In order to control the interrupts generated by this A/D Converter
;   this routine must place the address of its interrupt service into
;   the interrupt vector specified by this card.
;
;   The first location (usually 400) is the normal interrupt service
;   routine's address.  The next is the desired CPU status word.
;
;   The INVEC+4 is the address of the Error interrupt service while
;   the following address is its CPU status word.
;
        MOV     #FADIS,   @ #INVEC      ; Set I.S. address (normal)
        MOV     #003400,  @ #INVEC+2    ; Block other interrupts
        MOV     #FADER,   @ #INVEC+4    ; Set I.S. address (error)
        MOV     #003400,  @ #INVEC+6    ; Block other interrupts
;
;   Here the Data array pointer PDATA and the Data count DCNT are
;   reset.  This allows the routine to be restarted.
;
        MOV     #DATA, PDATA    ; Set Data array pointer.
        MOV     #100., DCNT     ; Get actual Data count.
;
        .SBTTL  Setup Channel & Gain Array
;
;   This section sets up the channel file array.  This array specifies
;   which multiplexor channels are to be selected and then converted.
;
;   First set up Preset pointers, these indicate which portion of
;   the channel array is to be used.  This example uses the first
;   four bytes of the first page of the channel file.
;
        MOVB    #053, @ #ADCSR + 1      ; Pick start address, mode 01
        MOVB    #000, @ #CFPR + 1       ; set initial pointer to zero.
;
        MOVB    #063, @ #ADCSR + 1      ; Pick final address, mode 01
        MOVB    #003, @ #CFPR + 1       ; set end pointer to three.
;
;   First select channel array page zero and starting word 0
;
        MOV     #002400, @ #ADCSR       ; page 00, mode 10 and byte 0
;
;   Write each channel and gain byte into the first four locations.
;
        MOV     #GANDC, RO              ; RO points to channel array
        MOV     #4, R1                  ; R1 has channel array count.
CLOOP:  MOVB    (RO)+, @ #CFPR          ; Move channel and Gain value.
        SOB     R1, CLOOP               ; If count not zero get next
```

```
;
;  Now Preload and Enable the Multiplexer logic.
;
        MOVB    #006, @ #ADCSR + 1      ; Preload Multiplexer, mode 11
        MOVB    #000, @ #CFPR           ; Give it a kick.
        MOVB    #000, @ #ADCSR + 1      ; Enable counters, mode 00
;
;  Now set DMACSR to indicate use of Programed I/O method.
;
        MOVB    #000, @ #DMACSR         ; Select Programmed I/O mode.
;
;
;  Now start converter by setting the external trigger enable
;  bit in ADCSR.  Note that the other control bits like channel
;  page select, error interrupt enable and done interrupt enable
;  must also be set correctly.
;
;  Binary: 0 100 000 001 001 000  or Octal: 040110
;
        MOV     #040110, @ #ADCSR       ; Start converter.
;
;  Now we return to calling routine and wait for an interrupt.
;
        RTS     PC      ; Back to calling routine.
;
;
        .SBTTL Interrupt Service Subroutine.
;
;  This code will be executed after each conversion is completed.
;
FADIS:  MOV     @ #ADBUF, @ PDATA       ; Move value into data buffer
        INC     PDATA                   ; Increment Data pointer
        INC     PDATA                   ; to the next word.
        DEC     DCNT                    ; Decrement Data Count
        BLE     DONE                    ; If last point go to DONE
;
;  If this is not the last point return
;  from the interrupt and wait for the next conversion to complete.
;
        RTI                             ; return from interrupt
;
;  When the Data count goes to zero all the data has been collected.
;  This routine moves the Control & Status Register's value over to
;  a status word in this routine to signal that all the data has
;  been collected.
;
;  Mark ISTATUS with new values.
;
DONE:   MOV     @ #ADCSR, @ PSTAT       ; Record converter's status
        CLR     @ #ADCSR                ; Shut down converter
        RTI                             ; Return from interrupt
;
;  The following code is only executed if the converter detects
;  an error.  This indicates that a data overrun has occured.
;
FADER:  MOV     @ #ADCSR, @ PSTAT       ; Record converter's status
        CLR     @ #ADCSR                ; Shut down converter
        RTI                             ; Return from interrupt
        .END
        .ENABLE LC
        .TITLE FADEX3  Fast A/D (DT3362) Programming Example 3
;
        .SBTTL Functional description
;
;   Tested: 25 Mar 81
;
;   This routine is a programming example designed to operate the
;   Data Translation Fast A/D converter: DT3362 .
;
;   In this example the Converter is required to make one hundred
;   conversions in DMA Burst mode.  These conversions are intiated
;   by the software and will all be taken as quickly as the converter
;   can go. The multiplexer channels
;   and their gain values are specified in the data block below.
;
;   In this example the channel array specifies that channel #5
;   will be read with a gain of 1, then again with a gain of 2,
;   then channel #7 will be read with a gain of 1, and finally
;   channel #2 will be read with a gain of one.
;
;   The interrupt service routine appears at the bottom
;   of this listing.
;
;
        .SBTTL  A/D data block, and control block definitions.
;
BASE   = 170400          ; Base address of Fast A/D converter
;
ADCSR  = BASE            ; A/D Control and Status Register
ADBUF  = BASE+2          ; A/D Data Buffer Register (not used)
CFPR   = BASE+4          ; Channel File Programming
DMACSR = BASE+6          ; DMA Control & Status
DMAWCR = BASE+10         ; DMA Word Count Register.
DMACAR = BASE+12         ; DMA Current Address Register.
;
INVEC  = 400             ; Address of Interrupt vector.
;
;  Gain & channel selection array follows
;
GANDC:  .BYTE   005      ; Gain 1 (code 0), channel 5
        .BYTE   105      ; Gain 2 (code 1), channel 5
        .BYTE   007      ; Gain 1 (code 0), channel 7
        .BYTE   002      ; Gain 1 (code 0), channel 2
;
;
;  The Data arrays and pointers.
;
PDATA:  DATA             ; Pointer to Data array. (100 values)
DATA:   .BLKW 100        ; Data array.
;
DCNT:   100.             ; Data count. (number of conversions)
;
PSTAT:  STATS            ; Pointer to status word array. (2 values)
STATS:  .BLKW 2          ; Space for two status words.
PSTAT:  .BLKW   1        ; Pointer to status word array. (2 values)
```

```
;
FADE3::                 ; This is the routines entry point.
;
        .SBTTL  Initialization (in case of powerup or reset)
;
;    This initialization code need only be executed once after
;    a powerup or hardware reset.
;
        MOV     #001400, @#ADCSR        ; select write to channel file
        MOVB    #002, @#CFPR+1          ; write required setup code.
;
        .SBTTL  Setup the interrupt vector
;
;   In order to control the interrupts generated by this A/D Converter
;   this routine must place the address of its interrupt service into
;   the interrupt vector specified by this card.
;
;   The first location (usually 400) is the normal interrupt service
;   routine's address.  The next is the desired CPU status word.
;
;   The INVEC+4 is the address of the Error interrupt service while
;   the following address is its CPU status word.
;
        MOV     #FADIS,  @ #INVEC       ; Set I.S. address (normal)
        MOV     #003400, @ #INVEC+2     ; Block other interrupts
        MOV     #FADER,  @ #INVEC+4     ; Set I.S. address (error)
        MOV     #003400, @ #INVEC+6     ; Block other interrupts
;
;
        .SBTTL  Setup Channel & Gain Array
;
;   This section sets up the channel file array.  This array specifies
;   which multiplexor channels are to be selected and then converted.
;
;   First set up Preset pointers, these indicate which portion of
;   the channel array is to be used.  This example uses the first
;   four bytes of the first page of the channel file.
;
        MOVB    #053, @ #ADCSR + 1      ; Pick start address, mode 01
        MOVB    #000, @ #CFPR + 1       ; set initial pointer to zero.
;
        MOVB    #063, @ #ADCSR + 1      ; Pick final address, mode 01
        MOVB    #003, @ #CFPR + 1       ; set end pointer to three.
;
;   First select channel array page zero and starting word 0
;
        MOV     #002400, @ #ADCSR       ; Page 00, mode 10 and byte 0
;
;   Write each channel and gain byte into the first four locations.
;
        MOV     #GANDC, R0              ; R0 points to channel array
        MOV     #4, R1                  ; R1 has channel array count.
CLOOP:  MOVB    (R0)+, @ #CFPR          ; Move channel & gain
                                        ; value to board.
        SOB     R1, CLOOP               ; If not zero get next byte.
```

```
;
;  Now Preload and Enable the Multiplexer logic.
;
        MOVB    #006, @ #ADCSR + 1      ; Preload Multiplexer, mode 11
        MOVB    #000, @ #CFPR           ; Give it a kick.
        MOVB    #000, @ #ADCSR + 1      ; Enable counters, mode 00
;
;  Now set DMACSR to indicate use of DMA Burst Mode over the Q Bus.
;  The desired Byte in binary is: 00 000 101  or 005 octal
;
        MOVB    #005, @ #DMACSR         ; Select DMA Burst on Q bus.
;
;  Now set the DMA Current Address Register to the beginning of the
;  data array.
;
        MOV     PDATA, @ #DMACAR        ; Set DMA Address register.
;
;  Now set the DMA Word Count register.  Set to conversion count.
;
        MOV     DCNT, @ #DMAWCR         ; Set DMA Word Count reg.
        NEG     @ #DMAWCR               ; Set to 2's complement.
;
;
;  Now start converter by setting the start bit in ADCSR.  Note the
;  other control bits like channel page select, error interrupt enable
;  and done interrupt enable must also be set correctly.
;  The binary word is: 0 100 000 001 000 001  or 040101 octal
;
        MOV     #040101, @ #ADCSR       ; Start converter.
;
;  Now we return to calling routine and wait for an interrupt.
;
        WAIT            ; temorary stall to hold up CPU
        RTS     PC      ; Back to calling routine.
;
;
```

```
        .SBTTL Interrupt Service Subroutine.
;
;   This code will be executed after the entire burst of conversions
;   is completed.
;
;   When the Data count goes to zero all the data has been collected.
;   This routine moves the two Status words on the converter over to
;   status words in this routine to signal that all the data has
;   been collected.
;
;
FADIS:  MOV     @ #ADCSR, @ PSTAT           ; Record converter's status
        CLR     @ #ADCSR                    ; Shut down converter
        ADD     #002, PSTAT                 ; Bump pointer to next word
        MOV     @ #DMACSR, @ PSTAT          ; Record DMA status.
        RTI                                 ; Return from interrupt
;
;   The following code is only executed if the converter detects
;   an error.  This indicates that a data overrun has occured.
;
FADER:  MOV     @ #ADCSR, @ PSTAT           ; Record converter's status
        CLR     @ #ADCSR                    ; Shut down converter
        ADD     #002, PSTAT                 ; Bump pointer to next word
        MOV     @ #DMACSR, @ PSTAT          ; Record DMA status.
        RTI                                 ; Return from interrupt
;
        .END
        .ENABLE LC
        .TITLE FADEX4  Fast A/D (DT3362) Programming Example 4
;
        .SBTTL Functional description
;
;       Tested:  10 Mar 81
;
;   This routine is a programming example designed to operate the
;   Data Translation Fast A/D converter: DT3362 .
;
;   In this example the Converter is required to make one hundred
;   conversions in DMA mode running point by point.  This method
;   allows the user to control the exact interval between conversions
;   by controlling the rate of the external trigger.
;   The multiplexer channels
;   and their gain values are specified in the data block below.
;
;   In this example the channel array specifies that channel #5
;   will be read with a gain of 1, then again with a gain of 2,
;   then channel #7 will be read with a gain of 1, and finally
;   channel #2 will be read with a gain of one.
;
;   The interrupt service routine appears at the bottom
;   of this listing.
;
;
        .SBTTL  A/D data block, and control block definitions.
;
BASE   = 170400         ; Base address of Fast A/D converter
;
ADCSR  = BASE           ; A/D Control and Status Register
ADBUF  = BASE+2         ; A/D Data Buffer Register (not used)
CFPR   = BASE+4         ; Channel File Programming
DMACSR = BASE+6         ; DMA Control & Status
DMAWCR = BASE+10        ; DMA Word Count Register.
DMACAR = BASE+12        ; DMA Current Address Register.
;
INVEC  = 400            ; Address of Interrupt vector.
;
;   Gain & channel selection array follows
;
GANDC:  .BYTE   005     ; Gain 1 (code 0), channel 5
        .BYTE   105     ; Gain 2 (code 1), channel 5
        .BYTE   007     ; Gain 1 (code 0), channel 7
        .BYTE   002     ; Gain 1 (code 0), channel 2
;
;
;   The Data arrays and pointers.
;
PDATA:  DATA            ; Pointer to Data array. (100 values)
DATA:   .BLKW 100       ; Data array.
;
DCNT:   100.            ; Data count. (number of conversions)
;
PSTAT:  STATS           ; Pointer to status word array. (2 values)
STATS:  .BLKW 2         ; Space for two status words.
;
;
FADE4::                 ; This is the routines entry point.
;
```

```
;
        .SBTTL  Initialization (in case of powerup or reset)
;
;   This initialization code need only be executed once after
;   a powerup or hardware reset.
;
        MOV     #001400, @#ADCSR        ; select write to channel file
        MOVB    #002, @#CFPR+1          ; write required setup code.
;
;
        .SBTTL  Setup the interrupt vector
;
;  In order to control the interrupts generated by this A/D Converter
;  this routine must place the address of its interrupt service into
;  the interrupt vector specified by this card.
;
;  The first location (usually 400) is the normal interrupt service
;  routine's address.  The next is the desired CPU status word.
;
;  The INVEC+4 is the address of the Error interrupt service while
;  the following address is its CPU status word.
;
        MOV     #FADIS,  @ #INVEC       ; Set I.S. address (normal)
        MOV     #003400, @ #INVEC+2     ; Block other interrupts
        MOV     #FADER,  @ #INVEC+4     ; Set I.S. address (error)
        MOV     #003400, @ #INVEC+6     ; Block other interrupts
;
;
;               .SBTTL  Setup Channel & Gain Array
;
;  This section sets up the channel file array.  This array specifies
;  which multiplexor channels are to be selected and then converted.
;
;  First set up Preset pointers, these indicate which portion of
;  the channel array is to be used.  This example uses the first
;  four bytes of the first page of the channel file.
;
        MOVB    #053, @ #ADCSR + 1      ; Pick start address, mode 01
        MOVB    #000, @ #CFPR + 1       ; set initial pointer to zero.
;
        MOVB    #063, @ #ADCSR + 1      ; Pick final address, mode 01
        MOVB    #003, @ #CFPR + 1       ; set end pointer to three.
;
;  First select channel array page zero and starting word 0
;
        MOV     #002400, @ #ADCSR       ; page 00, mode 10 and byte 0
;
;  Write each channel and gain byte into the first four locations.
;
        MOV     #GANDC, R0              ; R0 points to channel array
        MOV     #4, R1                  ; R1 has channel array count.
CLOOP:  MOVB    (R0)+, @ #CFPR          ; Move channel & gain
                                        ; value to board.
        SOB     R1, CLOOP               ; If not zero get next byte.
;
;  Now Preload and Enable the Multiplexer logic.
;
        MOVB    #006, @ #ADCSR + 1      ; Preload Multiplexer, mode 11
        MOVB    #000, @ #CFPR           ; Give it a kick.
        MOVB    #000, @ #ADCSR + 1      ; Enable counters, mode 00
;
```

```
;
        .SBTTL  Setup DMA registers.
;
;   Now set DMACSR to indicate use of DMA point by point (neither
;   type of Burst mode) over the Q Bus.
;   The desired Byte in binary is: 00 000 001  or 001 octal
;
        MOVB    #001, @ #DMACSR         ; DMA point by point, Q bus.
;
;   Now set the DMA Current Address Register to the beginning of the
;   data array.
;
        MOV     PDATA, @ #DMACAR        ; Set DMA Address register.
;
;   Now set the DMA Word Count register to the 2's complement of
;   the conversion count.
;
        MOV     DCNT, @ #DMAWCR         ; Get the Word Count.
        NEG     @ #DMAWCR               ; Form 2's complement.
;
;
        .SBTTL  Converter Startup
;
;   Now ready the converter by setting the external trigger enable
;   bit in the ADCSR.  Note that the
;   other control bits like channel page select, error interrupt enable
;   and done interrupt enable must also be set correctly.
;   The binary word is: 0 100 000 001 001 000  or 040110 octal
;
        MOV     #040110, @ #ADCSR       ; Start converter.
;
;   Now we return to calling routine and wait for an interrupt.
;
        RTS     PC      ; Back to calling routine.
;
;
        .SBTTL Interrupt Service Subroutine.
;
;   This code will be executed after the data is collected.
;
;   This routine moves the two Status words on the converter over to
;   the status array in this routine to signal that all the data has
;   been collected.
;
;
FADIS:  MOV     @ #ADCSR, @ PSTAT       ; Record converter's status
        CLR     @ #ADCSR                ; Shut down converter
        ADD     #2, PSTAT               ; Bump pointer to next word
        MOV     @ #DMACSR, @ PSTAT      ; Record DMA status.
        RTI                             ; Return from interrupt
;
;   The following code is only executed if the converter detects
;   an error.  This indicates that a data overrun has occured.
;
FADER:  MOV     @ #ADCSR, @ PSTAT       ; Record converter's status
        CLR     @ #ADCSR                ; Shut down converter
        ADD     #2, PSTAT               ; Bump pointer to next word
        MOV     @ #DMACSR, @ PSTAT      ; Record DMA status.
        RTI                             ; Return from interrupt
;
        .END
        .ENABLE LC
        .TITLE FADEX5  Fast A/D (DT3362) Programming Example 5
;
        .SBTTL Functional description
;
;       Tested: 11 Mar 81
;
;  This routine is a programming example designed to operate the
;  Data Translation Fast A/D converter: DT3362 .
;
;  In this example the Converter is required to make one hundred
;  DMA conversions in semi-burst mode.  This method
;  allows the user to control the exact interval between each set
;  of conversions by controlling the rate of the external trigger.
;
;  The multiplexer channels and their gain values are specified
;  in the data block below.  Here 4 conversions make up a set.
;
;  In this example the channel array specifies that channel #5
;  will be read with a gain of 1, then again with a gain of 2,
;  then channel #7 will be read with a gain of 1, and finally
;  channel #2 will be read with a gain of one.
;
;  The interrupt service routine appears at the bottom
;  of this listing.
;
;
        .SBTTL  A/D data block, and control block definitions.
;
BASE   = 170400         ; Base address of Fast A/D converter
;
ADCSR  = BASE           ; A/D Control and Status Register
ADBUF  = BASE+2         ; A/D Data Buffer Register (not used)
CFPR   = BASE+4         ; Channel File Programming
DMACSR = BASE+6         ; DMA Control & Status
DMAWCR = BASE+10        ; DMA Word Count Register.
DMACAR = BASE+12        ; DMA Current Address Register.
;
INVEC  = 400            ; Address of Interrupt vector.
;
;  Gain & channel selection array follows
;
GANDC:  .BYTE   005     ; Gain 1 (code 0), channel 5
        .BYTE   105     ; Gain 2 (code 1), channel 5
        .BYTE   007     ; Gain 1 (code 0), channel 7
        .BYTE   002     ; Gain 1 (code 0), channel 2
;
;  The Data arrays and pointers.
;
PDATA:  DATA            ; Pointer to Data array. (100 values)
DATA:   .BLKW 100       ; Data array.
;
DCNT:   100.            ; Data count. (number of conversions)
;
PSTAT:  STATS           ; Pointer to status word array. (2 values)
STATS:  .BLKW 2         ; Space for two status words.
;
FADE5::                 ; This is the routines entry point.
;
```

```
;
        .SBTTL  Initialization (in case of powerup or reset)
;
;   This initialization code need only be executed once after
;   a powerup or hardware reset.
;
        MOV     #001400, @#ADCSR        ; select write to channel file
        MOVB    #002, @#CFPR+1          ; write required setup code.
;
;
        .SBTTL  Setup the interrupt vector
;
;  In order to control the interrupts generated by this A/D Converter
;  this routine must place the address of its interrupt service into
;  the interrupt vector specified by this card.
;
;  The first location (usually 400) is the normal interrupt service
;  routine's address.  The next is the desired CPU status word.
;
;  The INVEC+4 is the address of the Error interrupt service while
;  the following address is its CPU status word.
;
        MOV     #FADIS,  @ #INVEC       ; Set I.S. address (normal)
        MOV     #003400, @ #INVEC+2     ; Block other interrupts
        MOV     #FADER,  @ #INVEC+4     ; Set I.S. address (error)
        MOV     #003400, @ #INVEC+6     ; Block other interrupts
;
;
        .SBTTL  Setup Channel & Gain Array
;
;  This section sets up the channel file array.  This array specifies
;  which multiplexor channels are to be selected and then converted.
;
;  First set up Preset pointers, these indicate which portion of
;  the channel array is to be used.  This example uses the first
;  four bytes of the first page of the channel file.
;
        MOVB    #053, @ #ADCSR + 1      ; Pick start address, mode 01
        MOVB    #000, @ #CFPR + 1       ; set initial pointer to zero.
;
        MOVB    #063, @ #ADCSR + 1      ; Pick final address, mode 01
        MOVB    #003, @ #CFPR + 1       ; set end pointer to three.
;
;  First select channel array page zero and starting word 0
;
        MOV     #002400, @ #ADCSR       ; page 00, mode 10 and byte 0
;
;  Write each channel and gain byte into the first four locations.
;
        MOV     #GANDC, R0              ; R0 points to channel array
        MOV     #4, R1                  ; R1 has channel array count.
CLOOP:  MOVB    (R0)+, @ #CFPR          ; Move channel & gain
                                        ; value to board.
        SOB     R1, CLOOP               ; If not zero get next byte.
;
;  Now Preload and Enable the Multiplexer logic.
;
        MOVB    #006, @ #ADCSR + 1      ; Preload Multiplexer, mode 11
        MOVB    #000, @ #CFPR           ; Give it a kick.
        MOVB    #000, @ #ADCSR + 1      ; Enable counters, mode 00
;
```

```
;
        .SBTTL  Setup DMA registers.
;
;   Now set DMACSR to indicate use of DMA semi-burst over the Q-bus
;   The desired Byte in binary is: 00 001 001  or 011 octal
;
        MOVB    #011, @ #DMACSR         ; DMA semi-burst Q bus.
;
;   Now set the DMA Current Address Register to the beginning of the
;   data array.
;
        MOV     PDATA, @ #DMACAR        ; Set DMA Address register.
;
;   Now set the DMA Word Count register to the 2's complement of
;   the conversion count.
;
        MOV     DCNT, @ #DMAWCR         ; Get the Word Count.
        NEG     @ #DMAWCR               ; Form 2's complement.
;
;
        .SBTTL  Converter Startup
;
;   Now ready the converter by setting the external trigger enable
;   bit in the ADCSR.  Note that the
;   other control bits like channel page select, error interrupt enable
;   and done interrupt enable must also be set correctly.
;   The binary word is: 0 100 000 001 001 000  or 040110 octal
;
        MOV     #040110, @ #ADCSR       ; Start converter.
;
;   Now we return to calling routine and wait for an interrupt.
;
        WAIT            ; wait for interrupt before returning.
        RTS     PC      ; Back to calling routine.
;
;
        .SBTTL Interrupt Service Subroutine.
;
;   This code will be executed after the converter stores all the data.
;
;   This routine moves the two Status array on the converter over to
;   a status array in this routine to signal that all the data has
;   been collected.
;
;
FADIS:  MOV     @ #ADCSR, @ PSTAT       ; Record converter's status
        CLR     @ #ADCSR                ; Shut down converter
        ADD     #2, PSTAT               ; Bump pointer to next word
        MOV     @ #DMACSR, @ PSTAT      ; Record DMA status.
        RTI                             ; Return from interrupt
;
;   The following code is only executed if the converter detects
;   an error.  This indicates that a data overrun has occured.
;
FADER:  MOV     @ #ADCSR, @ PSTAT       ; Record converter's status
        CLR     @ #ADCSR                ; Shut down converter
        ADD     #2, PSTAT               ; Bump pointer to next word
        MOV     @ #DMACSR, @ PSTAT      ; Record DMA status.
        RTI                             ; Return from interrupt
;
```

CHAPTER 9

A/D CALIBRATION

9.1 INTRODUCTION

This section contains calibration procedures for the data acquisition modules of the DT3362 series A/D conversion boards. The data acquisition module is the module next to the top edge of the DT3362 series board and is marked by its model number: DT5712, DT5725, DT5727, DT5704, or DT5726. Small holes leading to 15-turn potentiometers can be seen on each module's edge. The DT5712, DT5722, DT5725, and DT5727 contain three potentiometers. They are labeled FULL SCALE, PG ZERO, and ZERO. The DT5704 and DT5726 contain only two potentiometers labeled FULL SCALE and ZERO. The calibration is performed by adjusting these potentiometers. Table 9-1 shows the A/D converter modules present on each model of the DT3362 series boards.

TABLE 9-1: DT3362 SERIES A/D CONVERTER MODULES

MODEL	A/D MODULE USED
DT3362-16SE/8DI	DT5712
DT3362-32DI/64SE	DT5712
DT3362-16SE/8DI-PGH	DT5712
DT3362-64SE/32DI-PGH	DT5712
DT3362-F-16SE	DT5725
DT3362-F-8DI	DT5725
DT3362-F-64SE	DT5725
DT3362-F-32DI	DT5725
DT3362-G-16SE	DT5727
DT3362-G-8DI	DT5727
DT3362-G-64SE	DT5727
DT3362-G-32DI	DT5727
DT3362-H-16SE	DT5722
DT3362-H-8DI	DT5722
DT3362-H-64SE	DT5722
DT3362-H-32DI	DT5722
DT3362-H-16SE-PGH	DT5722
DT3362-H-8DI-PGH	DT5722
DT3362-H-64SE-PGH	DT5722
DT3362-H-32DI-PGH	DT5722
DT3377	DT5726
DT3368-4SE	DT5704
DT3368-12SE	DT5704

9.2 GUIDELINES

This section discusses general guidelines which should be observed when performing a calibration on the A/D module of a DT3362 series board.

9.2.1 PREREQUISITES

1. Calibration is done on Channel 0 with all other channels returned to Analog Ground.
2. The input cable is less than 3.0 meters (9.8 ft) long.
3. Software is available to provide continuous readings of Channel 0 and display them on a CRT.

9.2.2 EQUIPMENT REQUIRED

1. Precision voltage source, Electronic Development Corporation model 501J, or equivalent.
2. Extender card
3. Small, flat-bladed screwdriver
4. Miscellaneous cables and connectors

9.2.3 PRECAUTIONS

1. Switch the computer power and the DT3362 series board power off when installing or removing the DT3362 series board.
2. Make input connections only to a powered board.

9.2.4 SINGLE-ENDED INPUT OPERATION (SE)

1. In the SE mode, all the analog inputs are referenced to a single ground potential.
2. Connections are made as indicated in Figures 9-1 (all models except DT3368-4SE and DT3368-12SE) and 9-2 (models DT3368-4SE and DT3368-12SE).

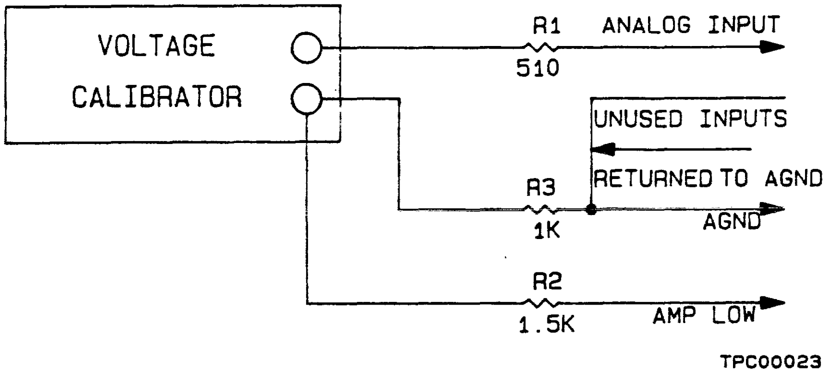


FIGURE 9-1: SINGLE-ENDED CONNECTIONS
(EXCEPT DT3368-4SE AND DT3368-12SE)

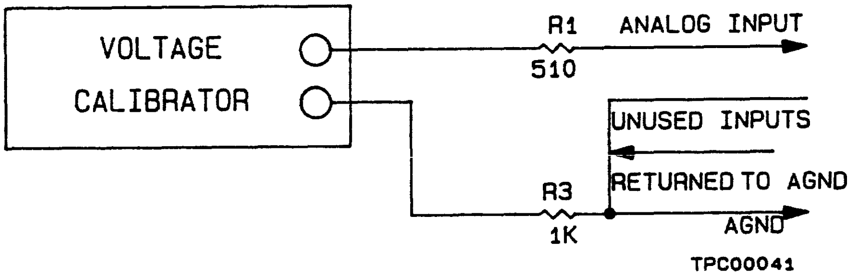


FIGURE 9-2: SINGLE-ENDED CONNECTIONS
(DT3368-4SE AND DT3368-12SE)

3. R1 is used for additional input protection.

4. R2 is used to balance the amplifier input and represents 1 kilohm for the multiplexer plus 510Œ for the input protection resistor R1.

5. R3 isolates the calibrator lo signal from the analog ground.

9.2.5 DIFFERENTIAL INPUT OPERATION (DI)

1. In the differential mode of operation each input has a high (Ch 0) and a low (Ch 0 Ret) connection (Figure 9-2). The measurement taken is the difference between the two inputs, thereby rejecting any noise common to both inputs.

2. With non-isolated systems, care must be taken to ensure that the inputs are referenced to Analog Ground.

3. If the signal to be measured is isolated, a return path for the bias current must be provided back to Analog Ground. This may be done by connecting a 1 to 100 kilohm resistor from the signal low to Analog Ground.

4. If all analog inputs are referenced to a single Analog Ground, only one resistor will be required between the external Analog Common and the board Analog Common.

5. A direct connection could be made between signal low and Analog Ground. However, this could cause large ground currents to flow if the computer ground potential is several volts different from the input signal ground potential. Even if the signal inputs are isolated from earth ground, ac currents could flow due to stray capacitance.

6. Connections are made as indicated in Figure 9-2.

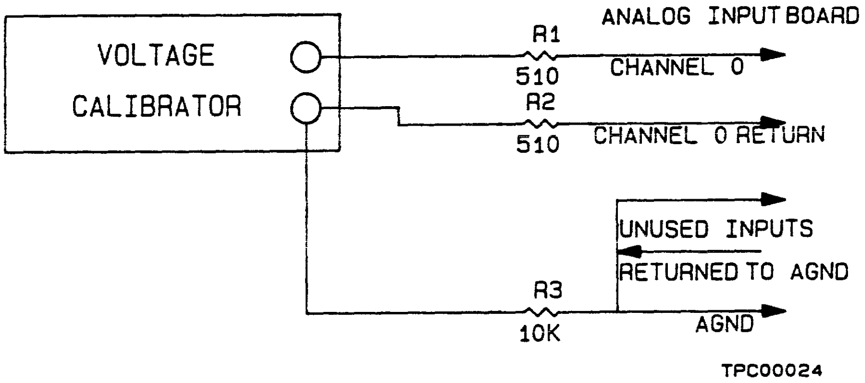


FIGURE 9-2: DIFFERENTIAL CONNECTIONS

7. R1 and R2 are used for additional input protection.

8. R3 is used to reference the inputs to Analog Ground. Without R3, input bias currents can build an input voltage exceeding the common mode voltage range.

9.3 CALIBRATION

Read the following notes before you start the calibration. Use the tables contained in these notes to obtain the values of the calibration parameters.

1. Pull the DT3362 series board out of the cage and insert an extender card into the backplane slot occupied by the board. Then connect the board to the extender. This will provide access to the potentiometers on the outer edge of the module.

2. Power up the system--the host Q-bus computer and the DT3362 series A/D board mounted on the computer's backplane. Allow one hour for the system to warm up and stabilize before calibration. The test equipment must also warm up before measurements are made.

3. The calibration must be done at a gain of 1 first.

4. When adjusting the potentiometers, center the code on the proper value by noting where the adjacent codes are. That is, mechanically center the potentiometer between the position that yields the desired value plus 1 LSB and the desired value minus 1 LSB.

5. A second calibration must be done to eliminate interaction between the Zero and Full Scale adjustments.

6. On a ±10 volt input, the range is 20V. (Range equals the upper input voltage minus the lower input voltage. For ±10V, that is +10V - (-10V) = 20V range.)

7. Table 9-2 lists the LSB voltage values for the 12-bit and 16-bit modules at different input ranges.

TABLE 9-2: 1 LSB VOLTAGE LEVEL

INPUT RANGE	12 BITS 1 LSB	16 BITS 1 LSB
±10V	4.88mV	305 μV
0 to +10V (±5V)	2.44mV	153 μV
0 to +5V (±2.5V)	1.22mV	76.5 μV
0 to +2.5V (±1.25V)	610 μV	38.2 μV
0 to +1.25V	305 μV	19.1 μV

8. Tables 9-3 and 9-4 give the minus full scale plus 1 LSB output codes for the 12-bit (offset binary and two's complement binary) and 16-bit (two's complement binary) resolution modules.

TABLE 9-3: 12-BIT OFFSET BINARY
AND TWO'S COMPLEMENT BINARY CODE
FOR MINUS FULL SCALE PLUS 1 LSB

OUTPUT CODE	MINUS FULL SCALE PLUS 1 LSB BINARY	HEX
Offset Binary	0000 0000 0001	001
Two's Complement	1000 0000 0001	801

TABLE 9-4: 16-BIT TWO'S COMPLEMENT BINARY CODE FOR MINUS FULL SCALE PLUS 1 LSB

TWO'S COMPLEMENT OUTPUT	MINUS FULL SCALE PLUS 1 LSB BINARY	HEX
16-bit	1000 0000 0000 0001	8001

9. Table 9-5 lists the plus full scale minus 2 LSB voltage values of the 12-bit and 16-bit modules at different input ranges.

TABLE 9-5: PLUS FULL SCALE MINUS 2 LSB

INPUT RANGE	12 BITS	16 BITS
±10V	+9.9902 V	+9.99939 V
0 to +10V	+9.9951 V	+9.99969 V
±5V	+4.9951 V	+4.99969 V
0 to +5V	+4.9976 V	+4.99985 V
±2.5V	+2.4976 V	+2.49985 V
0 to +2.5V	+2.4988 V	+2.49992 V
±1.25V	+1.2488 V	+1.24992 V
0 to +1.25V	+1.2494 V	+1.24996 V

10. Tables 9-6 and 9-7 give the plus full scale -2 LSB output codes for the 12-bit and 16-bit resolution modules.

TABLE 9-6: 12-BIT BINARY CODES FOR PLUS FULL SCALE MINUS 2 LSB

OUTPUT CODE	FULL SCALE -2 LSB BINARY	HEX
Straight Binary	1111 1111 1110	FFE
Offset Binary	1111 1111 1110	FFE
Two's Complement	0111 1111 1110	7FE

TABLE 9-7: 16-BIT TWO'S COMPLEMENT BINARY CODES
FOR PLUS FULL SCALE MINUS 2 LSB

RESOLUTION	PLUS FULL SCALE -2 LSB BINARY	 HEX
16-bit	0111 1111 1111 1110	7FFE

9.3.1 UNIPOLAR RANGE CALIBRATION

The unipolar range calibration requires the adjustment of three potentiometers:

- Unipolar Zero
- Unipolar Full Scale
- PG Zero

NOTE

The PG Zero calibration applies to modules DT5712 (-PGH version boards only), DT5725, and DT5727. The DT5704 does not contain a programmable gain amplifier and the DT5726 does not provide a user-adjustable programmable gain potentiometer.

9.3.1.1 Unipolar Zero

1. Input 1 LSB of signal (note Table 9-2) and adjust the Zero potentiometer for 1 count on the digital output.

NOTE

On higher gain ranges, especially with 16-bit modules, the input can be increased to 10 LSB's to center up the outputs around 10 LSB's due to the noise level.

9.3.1.2 Unipolar Full Scale

1. Input plus full scale minus 2 LSB's and set the Full Scale potentiometer for the proper code as indicated in Tables 9-6 and 9-7.

NOTE

> On higher gain ranges, especially with 16-bit modules, the input can be reduced 10 LSB's to center up the outputs around full scale minus 10 LSB's.

2. Repeat unipolar zero and unipolar full scale calibration to verify no interaction.

9.3.1.3 PG Zero

1. To calibrate the zero at a gain other than one, first calibrate the zero and full scale at a gain of 1.
2. Switch the gain to maximum (8) and apply an input of 1 LSB of the range. For example, at a gain of 8, the full scale range would be 1.25 volts. Thus the input would be 305μV.
3. Adjust the PG Zero potentiometer on the module for 1 digital count.

NOTE

> On higher gain ranges, especially with 16-bit modules, the input can be increased to 10 LSB's to center up the outputs around 10 LSB's.

4. Repeat unipolar zero and range calibration to verify no interaction.

9.3.2 BIPOLAR RANGE CALIBRATION

The bipolar range calibration requires the adjustment of three potentiometers:

- Bipolar Zero
- Bipolar Full Scale
- PG Zero

NOTE

The PG Zero calibration applies to modules DT5712 (-PGH version boards only), DT5725, and DT5727. The DT5704 does not contain a programmable gain amplifier and the DT5726 does not provide a user-adjustable programmable gain potentiometer.

9.3.2.1 Minus Full Scale

1. Table 9-8 gives the minus full scale plus 1 LSB voltage values of the DT5712, DT5725, DT5727, DT5726, and DT5704 A/D converter modules at different bipolar input ranges.

TABLE 9-8: MINUS FULL SCALE PLUS 1 LSB VOLTAGE

INPUT RANGE	12 BITS	16 BITS
±10V	-9.9951 V	-9.99969 V
±5V (1)	-4.9976 V	
±2.5V (1)	-2.4988 V	
±1.25V (1)	-1.2494 V	

NOTE:
1. These input ranges are not available on the DT5704 and DT5726 modules.

2. Input minus full scale plus 1 LSB voltage and adjust the zero potentiometer for the correct output code as indicated in Tables 9-9 and 9-10.

TABLE 9-9: 12-BIT OFFSET BINARY AND
TWO'S COMPLEMENT BINARY CODES
FOR MINUS FULL SCALE PLUS 1 LSB

OUTPUT CODE	MINUS FULL SCALE PLUS 1 LSB BINARY	HEX
Offset Binary	0000 0000 0001	001
Two's Complement	1000 0000 0001	801

TABLE 9-10: 16-BIT TWO'S COMPLEMENT BINARY CODES
FOR MINUS FULL SCALE PLUS 1 LSB

RESOLUTION	MINUS FULL SCALE PLUS 1 LSB BINARY	HEX
16-bit	1000 0000 0000 0001	8001

9.3.2.2 Plus Full Scale

1. Input plus full scale minus 2 LSB's and adjust the Full Scale potentiometer for the proper code as indicated in Tables 9-6 and 9-7.

2. On the higher gain ranges input can be reduced 10 LSB's to center the output around full scale minus 10 LSB's.

9.3.2.3 PG Zero

1. To calibrate the zero at a gain other than one, first calibrate the zero and full scale at a gain of 1.

2. At a gain of 1, apply 0 volts at the input and note the digital output. It should read the same as one of the output codes given in Table 9-11, ±1 LSB.

3. Switch the gain to maximum (8) and apply 0 volts at the input.

4. Adjust the PG Zero potentiometer on the module for one of the output codes given in Table 9-11, ±1 LSB. See specifications for accuracy.

Table 9-11: 12-BIT OFFSET BINARY AND TWO'S COMPLEMENT BINARY OUTPUT CODES FOR 0 VOLTS

SELECTED CODE	BINARY NOTATION
Offset Binary	1000 0000 0000
Two's Complement	0000 0000 0000

5. Repeat minus full scale and plus full scale calibration to verify no interaction.

9.4 APPLICATIONS

1. Noise is the major problem with A/D converters. This is why the accuracy specifications include an noise specification of .2 LSB rms at a gain of 1. In addition, the input amplifier has a noise specification (typically 3μV) that is multiplied by gain.

2. To determine the peak to peak spread of the noise to the 3-sigma point (99.7%), the rms value is multiplied by 6.

3. Table 9-12 gives typical count spreads due to noise at ±10V A/D range.

TABLE 9-12: TYPICAL COUNT SPREAD DUE TO NOISE

GAIN	12-BIT A/D	16-BIT A/D
1,2,4,8,	1.2	1.8

NOTE:
1. One count equals one LSB.

4. All cables must be routed away from switching power supplies and digital signals.

5. In high noise environments, the inputs must be differential with shielded twisted pairs for the signal.

6. The shield should only be connected at one end.

7. Unused inputs should be returned to Analog Ground.

8. RC filters can be added at the inputs to remove high frequency noise pickup.

9. Digital averaging of the data can be done simply by adding sixteen A/D conversions and then shifting right 4 times (dividing by 16). This is especially useful in minimizing the rms noise in high gain applications.

10. Input settling time can be a problem if the input cable is over 4.5 meters in length or the source impedance over 1 kilohm.

9.5 GROUNDING

9.5.1 DIFFERENTIAL INPUTS

1. When the differential input scheme is used, there are two switches per channel. Thus the number of channels is cut in half. The benefits are that common mode voltages, i.e., voltages appearing on both sides of the source simultaneously can be rejected by the differential input instrumentation amplifier. The amount of CMR depends on how well balanced the impedances are on the instrumentation amplifier inputs. (Typical spec. of 80dB at 60Hz with 1 kilohm unbalanced.)

2. Note that even though a differential input is used, the inputs must be referenced back to the Analog Ground of the system. This must be done because the signal plus common mode voltage range of the input is ±11 volts. If the input is not referenced back to Analog Ground, leakage currents on the input will drive the inputs out of the common mode voltage range, saturating the amplifier.

3. This ground reference need not be a hard connection but some impedance from 1 to 100 kilohms.

4. It is recommended that this connection be made with a 1 to 100 kilohm impedance rather than a direct connection to minimize potential ground current through the analog system. For example, the computer analog ground system eventually would be connected back to earth ground at a different location, which could result in several volts of ground potential between the two. If these two points are connected directly, several amps of current could flow through the analog path creating noisy data. If the connection is made through a 1 kilohm impedance, one volt ground difference will only cause one milliamp of current through the analog path thereby creating minimum disturbance to the analog measuring system. Note the connections indicated in Figure 9-2.

5. If all the analog inputs have a common connection, the resistor need only be installed once between the two commons.

9.5.2 SINGLE-ENDED INPUTS

1. With single-ended inputs it is recommended that the inputs be isolated from earth ground.
2. All the signal commons then would be connected separately to Analog Ground of the board.

9.5.3 PSEUDO-DIFFERENTIAL INPUTS

1. This mode of operation is similar to a single-ended operation--preserves the number of single-ended input channels--but allows one common mode voltage for all inputs.
2. The amplifier low input is used as a remote ground sense wire.
3. EXAMPLE:
   A metal panel with BNC connectors mounted on it can be used as a common ground. To prevent the ground from floating out of the common mode voltage range, a 1 kilohm resistor could be connected between the panel and Analog Ground. This problem may arise if the ground is floating, due to bias currents building a voltage exceeding the common mode. The amplifier low signal could then be connected to the panel to sense the panel ground. See Figure 9-1.

APPENDIX A

DT3362 SERIES JUMPER SUMMARY

A.1 INTRODUCTION

This appendix provides a summary of information on all jumpers used on the DT3362 series boards. For convenience of presentation, eight figures are used to cover all sixteen models included in the series. Each figure contains complete jumpering information for one pair of boards: the basic model and the parallel expanded version. The jumpers are listed in numerical order. The configuration of a jumper is indicated by "In", "Out", or "X". An "In" or "Out" entry indicates a permanent configuration for the jumper. Such a jumper is configured appropriately in the factory and is not user-reconfigurable. An "X" entry indicates that the jumper is user-configurable. In this case, the factory configuration is noted in parentheses.

TABLE A-1: DT3362-16SE/8DI AND DT3362-64SE/32DI JUMPERS

JUMPER	DT3362-16SE/8DI	DT3362-64SE/32DI
W1	In	In
W2	Out	Out
W12	X (Out)	X (Out)
W13	X (In)	X (In)
W21	Out	Out
W22	Out	Out
W25	Out	Out
W26	X (In)	X (In)
W27	X (In)	X (In)
W28	X (Out)	X (Out)
W29	In	In
W30	In	In
W31	In	In
W32	Out	Out
W33	In	In
W34	Out	Out
W35	Out	Out
W36	Out	Out
W37	X (In)	X (In)
W38	X (Out)	X (Out)
W39	X (Out)	X (Out)
W40	X (Out)	X (Out)
W41	X (In)	X (In)
W42	X (Out)	X (Out)
W43	X (In)	X (In)
W44	X (Out)	X (Out)
W45	X (In)	X (In)
W46	X (In)	X (In)
W47	X (In)	X (In)
W48	X (Out)	X (Out)
W49	X (Out)	X (Out)
W50	X (Out)	X (Out)
W51	X (Out)	X (Out)
W52	X (Out)	X (Out)
W53	X (Out)	X (Out)
W54	In	In
W55	Out	Out
W56	In	In
W57	Out	Out
W58	X (In)	X (In)
W59	X (Out)	X (Out)
W60	X (In)	X (In)
w3-w4	N/A	X (Out)
w4-w5	N/A	X (In)
w5-w6	N/A	X (Out)
w6-w7	N/A	X (In)
w14-w15	X (In)	X (In)
w14-w16	X (Out)	X (Out)
w16-w18	X (In)	X (In)
w23-w24	X (Out)	X (Out)

TABLE A-2: DT3362-16SE/8DI-PGH AND DT3362-64SE/32I-PGH JUMPERS

JUMPER	DT3362-16SE/8DI-PGH	DT3362-64SE/32DI-PGH
W1	In	In
W2	Out	Out
W12	X (Out)	X (Out)
W13	X (In)	X (In)
W21	In	In
W22	In	In
W25	Out	Out
W26	X (In)	X (In)
W27	X (In)	X (In)
W28	X (Out)	X (Out)
W29	In	In
W30	In	In
W31	In	In
W32	Out	Out
W33	In	In
W34	Out	Out
W35	Out	Out
W36	Out	Out
W37	X (In)	X (In)
W38	X (Out)	X (Out)
W39	X (Out)	X (Out)
W40	X (Out)	X (Out)
W41	X (In)	X (In)
W42	X (Out)	X (Out)
W43	X (In)	X (In)
W44	X (Out)	X (Out)
W45	X (In)	X (In)
W46	X (In)	X (In)
W47	X (In)	X (In)
W48	X (Out)	X (Out)
W49	X (Out)	X (Out)
W50	X (Out)	X (Out)
W51	X (Out)	X (Out)
W52	X (Out)	X (Out)
W53	X (Out)	X (Out)
W54	In	In
W55	Out	Out
W56	In	In
W57	Out	Out
W58	X (In)	X (In)
W59	X (Out)	X (Out)
W60	X (In)	X (In)
w3-w4	N/A	X (Out)
w4-w5	N/A	X (In)
w5-w6	N/A	X (Out)
w6-w7	N/A	X (In)
w14-w15	X (In)	X (In)
w14-w16	X (Out)	X (Out)
w16-w18	X (In)	X (In)
w23-w24	X (Out)	X (Out)

TABLE A-3: DT3362-F-16SE AND DT3362-F-64SE JUMPERS

JUMPER	DT3362-F-16SE	DT3362-F-64SE
W1	In	In
W2	Out	Out
W12	X (Out)	X (Out)
W13	X (Out)	X (Out)
W21	In	In
W22	In	In
W25	Out	Out
W26	X (In)	X (In)
W27	X (In)	X (In)
W28	X (Out)	X (Out)
W29	In	In
W30	In	In
W31	In	In
W32	Out	Out
W33	In	In
W34	Out	Out
W35	Out	Out
W36	Out	Out
W37	X (In)	X (In)
W38	X (Out)	X (Out)
W39	X (Out)	X (Out)
W40	X (Out)	X (Out)
W41	X (In)	X (In)
W42	X (Out)	X (Out)
W43	X (In)	X (In)
W44	X (Out)	X (Out)
W45	X (In)	X (In)
W46	X (In)	X (In)
W47	X (In)	X (In)
W48	X (Out)	X (Out)
W49	X (Out)	X (Out)
W50	X (Out)	X (Out)
W51	X (Out)	X (Out)
W52	X (Out)	X (Out)
W53	X (Out)	X (Out)
W54	In	In
W55	Out	Out
W56	In	In
W57	Out	Out
W58	X (In)	X (In)
W59	X (Out)	X (Out)
W60	X (In)	X (In)
w3-w4	N/A	X (Out)
w4-w5	N/A	X (In)
w5-w6	N/A	X (Out)
w6-w7	N/A	X (In)
w14-w15	X (In)	X (In)
w14-w16	X (Out)	X (Out)
w16-w18	X (In)	X (In)
w23-w24	X (Out)	X (Out)

TABLE A-4: DT3362-F-8DI AND DT3362-F-32DI JUMPERS

JUMPER	DT3362-F-8DI	DT3362-F-32DI
W1	In	In
W2	Out	Out
W12	X (Out)	X (Out)
W13	X (Out)	X (Out)
W21	In	In
W22	In	In
W25	Out	Out
W26	X (In)	X (In)
W27	X (In)	X (In)
W28	X (Out)	X (Out)
W29	In	In
W30	In	In
W31	In	In
W32	Out	Out
W33	In	In
W34	Out	Out
W35	Out	Out
W36	Out	Out
W37	X (In)	X (In)
W38	X (Out)	X (Out)
W39	X (Out)	X (Out)
W40	X (Out)	X (Out)
W41	X (In)	X (In)
W42	X (Out)	X (Out)
W43	X (In)	X (In)
W44	X (Out)	X (Out)
W45	X (In)	X (In)
W46	X (In)	X (In)
W47	X (In)	X (In)
W48	X (Out)	X (Out)
W49	X (Out)	X (Out)
W50	X (Out)	X (Out)
W51	X (Out)	X (Out)
W52	X (Out)	X (Out)
W53	X (Out)	X (Out)
W54	In	In
W55	Out	Out
W56	In	In
W57	Out	Out
W58	X (In)	X (In)
W59	X (Out)	X (Out)
W60	X (In)	X (In)
w3-w4	N/A	X (In)
w4-w5	N/A	X (Out)
w5-w6	N/A	X (In)
w6-w7	N/A	X (Out)
w14-w15	X (Out)	X (Out)
w14-w16	X (In)	X (In)
w16-w18	X (Out)	X (Out)
w23-w24	X (In)	X (In)

TABLE A-5: DT3362-G-16SE AND DT3362-G-64SE JUMPERS

JUMPER	DT3362-G-16SE	DT3362-G-64SE
W1	In	In
W2	Out	Out
W12	X (Out)	X (Out)
W13	X (Out)	X (Out)
W21	In	In
W22	In	In
W25	Out	Out
W26	X (In)	X (In)
W27	X (In)	X (In)
W28	X (Out)	X (Out)
W29	In	In
W30	In	In
W31	In	In
W32	Out	Out
W33	In	In
W34	Out	Out
W35	Out	Out
W36	Out	Out
W37	X (In)	X (In)
W38	X (Out)	X (Out)
W39	X (Out)	X (Out)
W40	X (Out)	X (Out)
W41	X (In)	X (In)
W42	X (Out)	X (Out)
W43	X (In)	X (In)
W44	X (Out)	X (Out)
W45	X (In)	X (In)
W46	X (In)	X (In)
W47	X (In)	X (In)
W48	X (Out)	X (Out)
W49	X (Out)	X (Out)
W50	X (Out)	X (Out)
W51	X (Out)	X (Out)
W52	X (Out)	X (Out)
W53	X (Out)	X (Out)
W54	In	In
W55	Out	Out
W56	In	In
W57	Out	Out
W58	X (In)	X (In)
W59	X (Out)	X (Out)
W60	X (In)	X (In)
w3-w4	N/A	X (Out)
w4-w5	N/A	X (In)
w5-w6	N/A	X (Out)
w6-w7	N/A	X (In)
w14-w15	X (Out)	X (Out)
w14-w16	X (In)	X (In)
w16-w18	X (Out)	X (Out)
w23-w24	X (In)	X (In)

TABLE A-6: DT3362-G-8DI AND DT3362-G-32DI JUMPERS

JUMPER	DT3362-G-8DI	DT3362-G-32DI
W1	In	In
W2	Out	Out
W12	X (Out)	X (Out)
W13	X (Out)	X (Out)
W21	In	In
W22	In	In
W25	Out	Out
W26	X (In)	X (In)
W27	X (In)	X (In)
W28	X (Out)	X (Out)
W29	In	In
W30	In	In
W31	In	In
W32	Out	Out
W33	In	In
W34	Out	Out
W35	Out	Out
W36	Out	Out
W37	X (In)	X (In)
W38	X (Out)	X (Out)
W39	X (Out)	X (Out)
W40	X (Out)	X (Out)
W41	X (In)	X (In)
W42	X (Out)	X (Out)
W43	X (In)	X (In)
W44	X (Out)	X (Out)
W45	X (In)	X (In)
W46	X (In)	X (In)
W47	X (In)	X (In)
W48	X (Out)	X (Out)
W49	X (Out)	X (Out)
W50	X (Out)	X (Out)
W51	X (Out)	X (Out)
W52	X (Out)	X (Out)
W53	X (Out)	X (Out)
W54	In	In
W55	Out	Out
W56	In	In
W57	Out	Out
W58	X (In)	X (In)
W59	X (Out)	X (Out)
W60	X (In)	X (In)
w3-w4	N/A	X (In)
w4-w5	N/A	X (Out)
w5-w6	N/A	X (In)
w6-w7	N/A	X (Out)
w14-w15	X (Out)	X (Out)
w14-w16	X (In)	X (In)
w16-w18	X (Out)	X (Out)
w23-w24	X (In)	X (In)

TABLE A-7: DT3362-H-16SE AND DT3362-H-64SE JUMPERS

JUMPER	DT3362-H-16SE	DT3362-H-64SE
W1	In	In
W2	Out	Out
W12	X (Out)	X (Out)
W13	X (Out)	X (Out)
W21	Out	Out
W22	Out	Out
W25	Out	Out
W26	X (In)	X (In)
W27	X (In)	X (In)
W28	X (Out)	X (Out)
W29	In	In
W30	In	In
W31	In	In
W32	Out	Out
W33	In	In
W34	Out	Out
W35	Out	Out
W36	Out	Out
W37	X (In)	X (In)
W38	X (Out)	X (Out)
W39	X (Out)	X (Out)
W40	X (Out)	X (Out)
W41	X (In)	X (In)
W42	X (Out)	X (Out)
W43	X (In)	X (In)
W44	X (Out)	X (Out)
W45	X (In)	X (In)
W46	X (In)	X (In)
W47	X (In)	X (In)
W48	X (Out)	X (Out)
W49	X (Out)	X (Out)
W50	X (Out)	X (Out)
W51	X (Out)	X (Out)
W52	X (Out)	X (Out)
W53	X (Out)	X (Out)
W54	In	In
W55	Out	Out
W56	In	In
W57	Out	Out
W58	X (In)	X (In)
W59	X (Out)	X (Out)
W60	X (In)	X (In)
w3-w4	N/A	X (Out)
w4-w5	N/A	X (In)
w5-w6	N/A	X (Out)
w6-w7	N/A	X (In)
w14-w15	X (Out)	X (Out)
w14-w16	X (In)	X (In)
w16-w18	X (Out)	X (Out)
w23-w24	X (In)	X (In)

TABLE A-8: DT3362-H-8DI AND DT3362-H-32DI JUMPERS

JUMPER	DT3362-H-8DI	DT3362-H-32DI
W1	In	In
W2	Out	Out
W12	X (Out)	X (Out)
W13	X (Out)	X (Out)
W21	Out	Out
W22	Out	Out
W25	Out	Out
W26	X (In)	X (In)
W27	X (In)	X (In)
W28	X (Out)	X (Out)
W29	In	In
W30	In	In
W31	In	In
W32	Out	Out
W33	In	In
W34	Out	Out
W35	Out	Out
W36	Out	Out
W37	X (In)	X (In)
W38	X (Out)	X (Out)
W39	X (Out)	X (Out)
W40	X (Out)	X (Out)
W41	X (In)	X (In)
W42	X (Out)	X (Out)
W43	X (In)	X (In)
W44	X (Out)	X (Out)
W45	X (In)	X (In)
W46	X (In)	X (In)
W47	X (In)	X (In)
W48	X (Out)	X (Out)
W49	X (Out)	X (Out)
W50	X (Out)	X (Out)
W51	X (Out)	X (Out)
W52	X (Out)	X (Out)
W53	X (Out)	X (Out)
W54	In	In
W55	Out	Out
W56	In	In
W57	Out	Out
W58	X (In)	X (In)
W59	X (Out)	X (Out)
W60	X (In)	X (In)
w3-w4	N/A	X (In)
w4-w5	N/A	X (Out)
w5-w6	N/A	X (In)
w6-w7	N/A	X (Out)
w14-w15	X (Out)	X (Out)
w14-w16	X (In)	X (In)
w16-w18	X (Out)	X (Out)
w23-w24	X (In)	X (In)

TABLE A-9: DT3362-H-16SE-PGH AND DT3362-H-64SE-PGH JUMPERS

JUMPER	DT3362-H-16SE-PGH	DT3362-H-64SE-PGH
W1	In	In
W2	Out	Out
W12	X (Out)	X (Out)
W13	X (Out)	X (Out)
W21	In	In
W22	In	In
W25	Out	Out
W26	X (In)	X (In)
W27	X (In)	X (In)
W28	X (Out)	X (Out)
W29	In	In
W30	In	In
W31	In	In
W32	Out	Out
W33	In	In
W34	Out	Out
W35	Out	Out
W36	Out	Out
W37	X (In)	X (In)
W38	X (Out)	X (Out)
W39	X (Out)	X (Out)
W40	X (Out)	X (Out)
W41	X (In)	X (In)
W42	X (Out)	X (Out)
W43	X (In)	X (In)
W44	X (Out)	X (Out)
W45	X (In)	X (In)
W46	X (In)	X (In)
W47	X (In)	X (In)
W48	X (Out)	X (Out)
W49	X (Out)	X (Out)
W50	X (Out)	X (Out)
W51	X (Out)	X (Out)
W52	X (Out)	X (Out)
W53	X (Out)	X (Out)
W54	In	In
W55	Out	Out
W56	In	In
W57	Out	Out
W58	X (In)	X (In)
W59	X (Out)	X (Out)
W60	X (In)	X (In)
w3-w4	N/A	X (Out)
w4-w5	N/A	X (In)
w5-w6	N/A	X (Out)
w6-w7	N/A	X (In)
w14-w15	X (Out)	X (Out)
w14-w16	X (In)	X (In)
w16-w18	X (Out)	X (Out)
w23-w24	X (In)	X (In)

TABLE A-10: DT3362-H-8DI-PGH AND DT3362-H-32DI-PGH JUMPERS

JUMPER	DT3362-H-8DI-PGH	DT3362-H-32DI-PGH
W1	In	In
W2	Out	Out
W12	X (Out)	X (Out)
W13	X (Out)	X (Out)
W21	In	In
W22	In	In
W25	Out	Out
W26	X (In)	X (In)
W27	X (In)	X (In)
W28	X (Out)	X (Out)
W29	In	In
W30	In	In
W31	In	In
W32	Out	Out
W33	In	In
W34	Out	Out
W35	Out	Out
W36	Out	Out
W37	X (In)	X (In)
W38	X (Out)	X (Out)
W39	X (Out)	X (Out)
W40	X (Out)	X (Out)
W41	X (In)	X (In)
W42	X (Out)	X (Out)
W43	X (In)	X (In)
W44	X (Out)	X (Out)
W45	X (In)	X (In)
W46	X (In)	X (In)
W47	X (In)	X (In)
W48	X (Out)	X (Out)
W49	X (Out)	X (Out)
W50	X (Out)	X (Out)
W51	X (Out)	X (Out)
W52	X (Out)	X (Out)
W53	X (Out)	X (Out)
W54	In	In
W55	Out	Out
W56	In	In
W57	Out	Out
W58	X (In)	X (In)
W59	X (Out)	X (Out)
W60	X (In)	X (In)
w3-w4	N/A	X (In)
w4-w5	N/A	X (Out)
w5-w6	N/A	X (In)
w6-w7	N/A	X (Out)
w14-w15	X (Out)	X (Out)
w14-w16	X (In)	X (In)
w16-w18	X (Out)	X (Out)
w23-w24	X (In)	X (In)

TABLE A-11: DT3377 JUMPERS

JUMPER	DT3377
W1	In
W2	Out
W12	Out
W13	Out
W25	In
W26	Out
W27	Out
W28	Out
W29	In
W30	In
W31	In
W32	In
W33	In
W34	In
W35	In
W36	In
W37	Out
W38	In
W39	X (Out)
W40	X (Out)
W41	X (In)
W42	X (Out)
W43	X (In)
W44	X (Out)
W45	X (In)
W46	X (In)
W47	X (In)
W48	X (Out)
W49	X (Out)
W50	X (Out)
W51	X (Out)
W52	X (Out)
W53	X (Out)
W54	In
W55	Out
W56	In
W57	Out
W58	X (In)
W59	X (Out)
W60	X (In)
w3-w4	Out
w4-w5	Out
w5-w6	Out
w6-w7	Out
w14-w15	Out
w14-w16	In
w16-w18	Out
w23-w24	Out

TABLE A-12: DT3368-4SE AND DT3368-12SE JUMPERS

JUMPER	DT3368-4SE	DT3368-12SE
W1	In	In
W2	Out	Out
W12	X (Out)	X (Out)
W13	X (In)	X (In)
W25	Out	Out
W26	X (In)	X (In)
W27	X (In)	X (In)
W28	X (Out)	X (Out)
W29	In	In
W30	In	In
W31	In	In
W32	Out	Out
W33	In	In
W34	Out	Out
W35	Out	Out
W36	Out	Out
W37	X (In)	X (In)
W38	X (Out)	X (Out)
W39	X (Out)	X (Out)
W40	X (Out)	X (Out)
W41	X (In)	X (In)
W42	X (Out)	X (Out)
W43	X (In)	X (In)
W44	X (Out)	X (Out)
W45	X (In)	X (In)
W46	X (In)	X (In)
W47	X (In)	X (In)
W48	X (Out)	X (Out)
W49	X (Out)	X (Out)
W50	X (Out)	X (Out)
W51	X (Out)	X (Out)
W52	X (Out)	X (Out)
W53	X (Out)	X (Out)
W54	In	In
W55	Out	Out
W56	In	In
W57	Out	Out
W58	X (In)	X (In)
W59	X (Out)	X (Out)
W60	X (In)	X (In)
w3-w4	N/A	Out
w4-w5	N/A	In
w5-w6	N/A	Out
w6-w7	N/A	In
w14-w15	Out	Out
w14-w16	Out	Out
w16-w18	In	In
w23-w24	Out	Out

APPENDIX B

DATA CODING TABLES

B.1 INTRODUCTION

The DT3362 series board's analog to digital (A/D) converter can convert the analog voltage at any one of its input channels into a digital code called an Analog Data Value.

Data Translation analog peripherals use two different digital coding schemes, depending on the type of converter used. 8, 10, and 12-bit converters use offset binary for bipolar voltage ranges and straight binary for unipolar voltage ranges. 16-bit converters, however, use two's complement coding and are limited in hardware to bipolar ranges.

To convert an Analog Data Value into a voltage value, it is necessary to know the number of codes (NOC) used by the A/D or D/A converter, as well as the high (PFS) and the low (MFS) voltage values of the analog range. PFS is plus full scale, which is the highest voltage of a given range (for example, +10V on a ±10V full scale range); MFS is minus full scale, which is the lowest voltage of a given bipolar range (for example, -10V on ±10V). Zero is the lowest value for unipolar ranges. NOC is determined by raising the number 2 to the power n, where n is the number of bits of resolution in the analog converter. Thus, for an 8-bit converter NOC equals 256; for a 10-bit converter, 1024; for a 12-bit converter, 4096; and for a 16-bit converter, 65536.

NOTE

> The highest possible positive value is actually one LSB less than PFS. However, PFS is the correct number to use in the equations which follow.

To calculate the input voltage detected by an A/D converter which uses binary or offset binary coding use the equation in Figure B-1. To calculate the input voltage to an A/D converter which uses two's complement binary coding use the equation in Figure B-2.

$$\boxed{\text{Analog Voltage} = \left[\text{ADV X } \frac{(\text{PFS} - \text{MFS})}{\text{NOC}} \right] + (\text{MFS})}$$

FIGURE B-1: CALCULATING THE INPUT VOLTAGE TO AN A/D CONVERTER USING BINARY OR OFFSET BINARY CODING

$$\boxed{\text{Analog Voltage} = \left[\text{ADV X } \frac{(\text{PFS} - \text{MFS})}{\text{NOC}} \right]}$$

FIGURE B-2: CALCULATING THE INPUT VOLTAGE TO AN A/D CONVERTER USING TWO'S COMPLEMENT CODING

The balance of this section consists of a series of tables indicating Analog Data Values for key points in the input ranges of the DT3362 series boards. The tables list analog voltages in one column and the corresponding binary code in another. For convenience the binary codes are represented as hexadecimal numbers.

TABLE B-1: 12-BIT A/D BIPOLAR CODING TABLE
(ALL MODELS EXCEPT DT3377)

% OF FULL SCALE	DIGITAL CODING	ANALOG VOLTAGES GAIN OF 1	GAIN OF 2	GAIN OF 4	GAIN OF 8
+FS - 1 LSB	0FFF	+9.9951V	+4.9976V	+2.4988V	+1.2494V
+FS - 2 LSBS	0FFE	+9.9902V	+4.9951V	+2.4976V	+1.2488V
+1/2 FS	0C00	+5.0000V	+2.5000V	+1.2500V	+0.6250V
+1 LSB	0801	+4.8828LV	+2.4414mV	+1.2207mV	+610.35μV
0	0800	0.0000V	0.0000V	0.0000V	0.0000V
-1 LSB	07FF	-4.8828mV	-2.4414mV	-1.2207mV	-610.35μV
-1/2 FS	0400	-5.0000V	-2.5000V	-1.2500V	-0.6250V
-FS + 1LSB	0001	-9.9951V	-4.9976V	-2.4988V	-1.2494V
-FS	0000	-10.0000V	-5.0000V	-2.5000V	-1.2500V

NOTE:
1. The DT3368-4SE and DT3368-12SE operate at gain =1 only.

TABLE B-2: 12-BIT A/D UNIPOLAR CODING TABLE
(ALL MODELS EXCEPT DT3377)

% OF FULL SCALE	DIGITAL CODING	ANALOG VOLTAGES GAIN OF 1	GAIN OF 2	GAIN OF 4	GAIN OF 8
+FS - 1 LSB	0FFF	+9.9976V	+4.9988V	+2.4994V	+1.2497V
+FS - 2 LSBS	0FFE	+9.9951V	+4.9976V	+2.4988V	+1.2494V
+3/4 FS	0C00	+7.5000V	+3.7500V	+1.8750V	+0.9375V
+1/2 FS	0800	+5.0000V	+2.5000V	+1.2500V	+0.6250V
+1/4 FS	0400	+2.5000V	+1.2500V	+.62500V	+0.3125V
0 + 1 LSB	0001	+2.4414mV	+1.2207mV	+610.36μV	+305.17μV
0	0000	0.0000V	0.0000V	0.0000V	0.0000V

NOTE:
1. The DT3368-4SE and DT3368-12SE operate at gain =1 only.

TABLE B-3: 16-BIT A/D BIPOLAR CODING (DT3377)

% OF FULL SCALE	DIGITAL CODING	ANALOG VOLTAGES AT GAIN OF 1
+FS - 1 LSB	7FFF	+9.999695V
+FS - 2 LSBS	7FFE	+9.999390V
+1/2 FS	4000	+5.000000V
+1 LSB	0001	+305.1758μV
0	0000	0.000000V
-1 LSB	FFFF	-305.1758μV
-1/2 FS	C000	-5.000000V
-FS + 1 LSB	8001	-9.999695V
-FS	8000	-10.00000V

APPENDIX C

TROUBLESHOOTING

C.1 TROUBLESHOOTING

This appendix provides some easy troubleshooting tips which you can use when your board ceases to operate. Go through the following checklist before calling Data Translation for assistance:

1. Make sure the DT3362 series board is in the factory-shipped configuration.

2. Make sure that the device address and the interrupt vector address are set properly on the board.

3. Check the +5 volt power supply. It must be between +4.75 and +5.25 volts.

4. Ensure that all board jumpers affecting the A/D range are in the proper locations.

5. Ensure that all cables are connected properly.

6. Ensure that no components are making contact to an adjacent board.

7. Make sure all unused channels are referenced to Analog Ground.

TROUBLESHOOTING

C.1.1 SYMPTOMS AND SOLUTIONS

C.1.1.1 No Board Response

1. Check the steps listed under Troubleshooting Guide.
2. Clean board connector fingers with a soft pencil eraser.

C.1.1.2 Digital Output Code In Plus Or Minus Full Scale Stops

1. Check for floating inputs as discussed in Section 4.6.3.
2. Make sure that the input is not greater than the range that the A/D is set for.
3. Make sure that none of the other inputs is above the plus and minus power supplies (±15 volts).

C.1.1.3 First Reading On A Channel Is Incorrect

1. Settling time problems due to source impedance over 1 kilohm.
2. Input settling time is budgeted from one linear region to another. If the sample and hold is saturated due to overvoltage on an open channel, the first reading on a valid channel will be in error.

C.1.1.4 Zero Or Full Scale Will Not Calibrate

The total adjustment range of the Zero and Full Scale potentiometers is approximately 50 LSB's. If Zero or Full Scale cannot be calibrated, consult Data Translation.

C.2 RMA NUMBER

If factory service is necessary, the Customer Service department at Data Translation, Inc. should be called [(617) 481-3700] for a Return Material Authorization (RMA) Number. All return shipments to Data Translation, Inc. must be marked with RMA numbers.

The defective board should be packed in the original shipping materials, if available and in good condition. The board must be wrapped in an electrically conductive plastic, and it should be handled with ground protection. A static electric discharge can destroy components on the board. Data Translation Customer Service Department can be reached by any of the of the means listed below.

Customer Service Department
Data Translation, Inc.
100 Locke Drive
Marlboro, MA 01752-1192
Tel: (617) 481-3700
Telex: 951646 (DATATRANS MARO)
Easylink 62825999

APPENDIX D

LISTING OF DT3362 SERIES DIAGNOSTICS

This appendix contains the listing of the DT3362 series diagnostics.

ABLE OF CONTENTS

```
     2-    7     General Information
     3-   42     List of changes in program
     4-   65     List of program macros
     5-   86     Test Parameter Block (TPB)
     8-  138     Initialization
     9-  204     Display Parameters
    10-  221     A/D section initialization routines
    11-  252     Error reporters
    12-  303     Model testing information
    13-  323     Logic Tests
    13-  324     Test 1:  BRPLY from all registers
    14-  375     Test 2: Bit test of CSR
    15-  409     Test 3: Bit test of CFPR
    18-  507     Test 4: Bit test of DMACSR
    19-  541     Test 5: Bit test of DMAWCR
    20-  574     Test 6: Bit test of DMAADR
    21-  606     Test 7: Check A/D START bit
    22-  647     Test 10: BINITL action
    23-  690     Test 11: Byte operation of ADCSR
    25-  746     Test 12: Test of mux/gain memory
    27-  810     Test 13: Bit test of mux/gain memory
    30-  906     Test 14: A/D START bit triggering
    32-  987     Test 15: Error bit operation
    35- 1109     Test 16: A/D DONE interrupt
    37- 1166     Test 17: DMA logic
    41- 1291     Test 20: DMA burst and semi-burst test
    44- 1422     Test 21: DMA data transfer
    47- 1541     Test 22: end of logic tests
    48- 1555     test 23: Check DMA timeout bit operation
    50- 1652     Calibration initialization
    54- 1828     Test 24: A/D Calibration
    55- 1861     Test 25: A/D data display under DMA
    57- 1916     Test 26: A/D input channel scan
    58- 1953     Test 27: A/D input gain/channel scan
    60- 2011     Test 30: A/D timing test under DMA
    62- 2076     Test 31: A/D DMA programmable channel scan
    65- 2169     Test 32: A/D DMA gain channel scan
    68- 2262     Test 33: A/D DMA timing check
    69- 2304     Test 34: User specified A/D DMA
```

```
                                                  .TITLE  DT3362  TST-11 MODULE
                                                  .IDENT  /V01.02/
000000                                            .PSECT  DT3362
                                                  .ENABL  LC
                        .NLIST  BIN
                ;
                        .SBTTL  General Information
                ;
                ; Copyright (C) 1981, 1982.  Data  Translation  Incorporated.
                ; All rights reserved. No part of this program or publication
                ; may be reproduced without the prior written  permission  of
                ; Data  Translation  Incorporated, 100 Locke Drive, Marlboro,
                ; Massachusetts 01752.
                ;
                ;
                ; The information in  this  document  is  subject  to  change
                ; without  notice and should not be construed as a commitment
                ; by  Data  Translation   Incorporated.    Data    Translation
                ; Incorporated  assumes no responsibility for any errors that
                ; may appear in this document.
                ;
                ;
                ; Data Translation cannot assume any responsibility  for  the
                ; use  of  any  portion  of  this software for other than its
                ; intended diagnostic purpose in calibrating and testing Data
                ; Translation   manufactured  analog  and  digital  interface
                ; boards.
                ;
                ; SP no.  SP-0044
                ; Version 01-02
                ;
                ; Phillip Martinez  30-Jul-81
                ; Phillip Martinez  16-Oct-81
                ; Phillip Martinez  15-Apr-82
                ;
                ; This test routine works to test the Data Translation
                ; DT 3362 A/D interface board.
                ;
                ;
                ;
```

```
        .SBTTL  List of changes in program
;
;
;       The differences between V01-02 and V01-01 are:
;
;       1).  A macro was added to clear the done bit
;
;       2).  Error bit checking for the pipeline was added.
;
;
;       The differences between V01-01 and V01-00 are:
;
;       1).  The self timing loop was changed from a multiply by
;            8 to a multiply by 32.
;
;
;       This test diagnostic was released at
;       revision V01-00
;
;
;
                                .LIST   BIN
```

```
                .SBTTL  List of program macros
        ;
        ; The following defines all of the macros that are resident
        ; exclusivly in this program.
        ;
        ;
        ;  This macro clears the done bit on the board by reading
        ;  the data out of the pipeline until the pipeline is empty.
        ;  Assumed is that R1 points to the base address of the board
        ;  at the time the macro is called.
        ;
                .MACRO  DCLEAR  A,B
                BR      B               ; branch to check if done present
        A:      TST     2(R1)           ; read data to clear done
        B:      TSTB    (R1)            ; check for done bit
                BMI     A               ; loop to read data
                .ENDM   DCLEAR
        ;
        ;
        ;
```

```
                                        .SBTTL  Test Parameter Block (TPB)
                                ;
                                ; Test-11 Declaration
                                ;
                                        .MCALL  TST11
        000000                          TST11
                                .LIST   ME
                                ;
                                ; EQUATED SYMBOLS
                                ;
                                ;       TERMINAL I/O PARAMETERS
                                ;
                177560          RCSR    =177560         ; RECEIVER STATUS REGISTER
                177562          RBUF    =177562         ; RECEIVER DATA BUFFER
                177564          XCSR    =177564         ; TRANSMITTER STATUS REGISTER
                177566          XBUF    =177566         ; TRANSMITTER DATA BUFFER
                                ;
                                ;       SPECIAL ASCII CHARACTERS
                                ;
                000003          CTRLC   =3              ; CONTROL-C
                000012          LF      =12             ; LINE FEED
                000014          FF      =14             ; FORM FEED
                000015          CR      =15             ; CARRIAGE RETURN
                000040          SPACE   =40             ; SPACE CODE
                000011          CTRLI   =11             ; CONTROL-I (TAB)
                000177          DEL     =177            ; DELETE CODE
                                ;
                                ;
                                ; PRIORITY 7 PSW
                000340          PR7     =340
                                ;
                                ; BIT DEFINITIONS
                                ;
                100000          BIT15   =100000
                040000          BIT14   =40000
                020000          BIT13   =20000
                010000          BIT12   =10000
                004000          BIT11   =4000
                002000          BIT10   =2000
                001000          BIT9    =1000
                000400          BIT8    =400
                000200          BIT7    =200
                000100          BIT6    =100
                000040          BIT5    =40
                000020          BIT4    =20
                000010          BIT3    =10
                000004          BIT2    =4
                000002          BIT1    =2
                000001          BIT0    =1
```

```
                                ; TST-11 PARAMETER STORAGE
                                ;
                                ;
        000514                  ODTACC  =514    ; ODT ACCUMULATOR
        000526                  ERRCNT  =526    ; # OF ERRORS
                                ;
                                ;
                                ; PARAMETER STORAGE FOR CODE MODULES
                                ;
        000540                  SWR     =540    ; SWITCH REGISTER
        000542                  BASE    =542    ; BASE ADDRESS
        000544                  VECTOR  =544    ; VECTOR ADDRESS
        000546                  DELAY   =546    ; DELAY COUNT
                                ;
                                ;
        000560                  TFLAG   =560                    ; flag word for tests
                                        .GLOBL  DMABUF          ; DMA data buffer
                                ;
                                ; Test Parameter Block
                                ;
                .NLIST  BIN
        ;
000000  TPB:    .WORD   PARAM           ; address of parameter
                                        ; print-out routine
000002          .BYTE   377             ; reserved
000003          .BYTE   34              ; # of tests
        ;
        ; Test Address Table for use by TST-11
        ;
000004          .WORD   TEST1,PR7
000010          .WORD   TEST2,PR7
000014          .WORD   TEST3,PR7
000020          .WORD   TEST4,PR7
000024          .WORD   TEST5,PR7
000030          .WORD   TEST6,PR7
000034          .WORD   TEST7,PR7
000040          .WORD   TEST10,PR7
000044          .WORD   TEST11,PR7
000050          .WORD   TEST12,PR7
000054          .WORD   TEST13,PR7
000060          .WORD   TEST14,PR7
000064          .WORD   TEST15,PR7
000070          .WORD   TEST16,PR7
000074          .WORD   TEST17,PR7
000100          .WORD   TEST20,PR7
000104          .WORD   TEST21,PR7
000110          .WORD   TEST22,PR7
000114          .WORD   TEST23,PR7
000120          .WORD   TEST24,0
000124          .WORD   TEST25,0
000130          .WORD   TEST26,0
000134          .WORD   TEST27,0
000140          .WORD   TEST30,0
000144          .WORD   TEST31,0
000150          .WORD   TEST32,0
000154          .WORD   TEST33,0
000160          .WORD   TEST34,0
                                        .LIST   BIN
                                        .SBTTL  Initialization
                                ;
                                ;
000164  012737  000000  000540  INIT:   MOV     #0,@#SWR          ; preset SWR
000172                                  PRINT   <# of A/D input channels (in octal): >
000242                                  GETOCT                    ; get octal input
000244  103421                          BCS     3$                ; CR - continue
000246                          1$:     CRLF                      ; Ask again
000250                                  PRINTC  <Must be between 1 and 100 >
000306  000726                          BR      INIT
                                ;
000310  113700  000514          3$:     MOVB    @#ODTACC,R0       ; get the input value
000314  105700                          TSTB    R0                ; zero value?
000316  001753                          BEQ     1$                ; no - skip
000320  122700  000100                  CMPB    #100,R0           ; check for out of limits
000324  103750                          BCS     1$                ; error
                                ;
000326  000300                          SWAB    R0                ; put channel in high byte
000330  042700  100377                  BIC     #100377,R0        ; adjust 4 bits
000334  010037  000540                  MOV     R0,@#SWR          ; preset SWR
000340                                  RELMOV  #PG,R1            ; query user
000340  010701                          MOV     PC,R1             ; MOVE PC TO R1
000342  062701  000164                  ADD     #PG-.,R1          ; ADD IN RUN-TIME OFFSET
000346  004767  000032                  CALL    QUERY
000352  103403                          BCS     5$                ; typed "N"
000354  152737  000200  000540          BISB    #200,@#SWR        ; set PG bit
000362  012737  170400  000542  5$:     MOV     #170400,@#BASE    ; set default address
000370  012737  000400  000544          MOV     #400,@#VECTOR     ; set default vector
000376  005037  000546                  CLR     @#DELAY           ; preset delay to max.
000402  000207                          RETURN                    ; all done
                                ;
                                ;
                                ; Software switch register bit reservations
                                ;
                                ;
                                ; bits 11-8:      # of A/D channels
                                ; bit 7:          PG option installed
                                ; bits 0-6:       not used
                                ;
                                ;
000404                          QUERY:  PUSH    R1              ; save pointer
000406                                  PRINTS                  ; print prompt
000410                                  PRINT   < (Y or N)? >
000426                                  TTYIN
000430  122700  000131                  CMPB    #'Y,R0          ; a "Y"?
000434  001427                          BEQ     3$
000436  122700  000116                  CMPB    #'N,R0          ; a "N"?
000442  001422                          BEQ     1$              ; yes
000444                                  CRLF
000446                                  PRINTC  <See manual for assistance.>
000504                                  POP     R1              ; restore pointer
000506  000736                          BR      QUERY
                                ;
000510  000261                  1$:     SEC                     ; set carry flag
000512  000401                          BR      5$
000514  000241                  3$:     CLC                     ; clear carry flag
000516                          5$:     TTYOUT                  ; echo character
000520                                  CRLF
000522                                  POP     R1              ; restore R1
000524  000207                          RETURN
                                ;
                .NLIST  BIN
000526  PG:     .ASCIZ  "Is PG option installed"
000555  PDTA:   .ASCIZ  "Print the data"
                .EVEN
                                        .LIST   BIN
                                ;
                                        .SBTTL  Display Parameters
                                ;
                                ;       This routine displays the current setting
                                ;       of 'BASE' and 'VECTOR' on the system console
                                ;       terminal.
                                ;
000574                          PARAM:  PRINT   <  Base address = >
000620  013700  000542                  MOV     @#BASE,R0       ; get base address
000624                                  OCT16                   ; display
000626                                  CRLF
000630                                  PRINT   <Vector address = >
000654  013700  000544                  MOV     @#VECTOR,R0     ; get vector address
000660                                  OCT16                   ; display
000662                                  CRLF
000664  000207                          RETURN                  ; all done
                                ;
```

```
                                                 .SBTTL  A/D section initialization routines
                                         ;
                                         ;
000666  112761  000003  000001  PTRINT:  MOVB    #3,1(R1)            ; program CFPR chip
000674  112761  000002  000005           MOVB    #2,5(R1)            ; write chip initialization
000702  000207                           RETURN                      ; return
                                         ;
                                         ;
000704  005011                  LSETUP:  CLR     (R1)                ; clear ADCSR
000706  005061  000006                   CLR     6(R1)               ; clear DMACSR
000712  004767  177750                   CALL    PTRINT              ; init. pointer chip
000716  112761  000053  000001           MOVB    #53,1(R1)           ; command to write first channel
000724  112761  000000  000005           MOVB    #0,5(R1)            ; first channel
000732  112761  000063  000001           MOVB    #63,1(R1)           ; command to write last channel
000740  112761  000000  000005           MOVB    #0,5(R1)            ; last channel
000746  112761  000005  000001           MOVB    #5,1(R1)            ; command to write channel address
000754  112761  000000  000004           MOVB    #0,4(R1)            ; write it
000762  112761  000043  000001  MXRST:   MOVB    #43,1(R1)           ; reset pointer
000770  112761  000000  000005           MOVB    #0,5(R1)            ; write strobe
000776  112761  000006  000001  MUXLD:   MOVB    #6,1(R1)            ; command to load mux's
001004  112761  000000  000004           MOVB    #0,4(R1)            ; strobe
001012  112761  000000  000001           MOVB    #0,1(R1)            ; do it
001020  000207                           RETURN                      ; return to test program
                                         ;
001022  112761  000053  000001  PTRRST:  MOVB    #53,1(R1)           ; write pointer write command
001030  112761  000000  000005           MOVB    #0,5(R1)            ; program starting address
001036  112761  000063  000001           MOVB    #63,1(R1)           ; write pointer write command
001044  112761  000377  000005           MOVB    #377,5(R1)          ; program end pointer
001052  000207                           RETURN                      ; return
                                         ;
```

```
                                        .SBTTL  Error reporters
                                ;
                                ;
                                ; This routine provides error reporting for bus
                                ; time-out errors (no BRPLY from interface).
                                ;
001054                          NORPLY: PRINT   <No BRPLY when accessing location >
001120  010100                          MOV     R1,R0
001122                                  OCT16                   ; display address
001124                                  CRLF
001126  000207                          RETURN                  ; done
                                ;
                                ;
                                ; This routine provides error reporting for register
                                ; bit errors (one or more incorrect bits in a register).
                                ;
001130                          REG:    PRINTC  <Register Error>
001152                                  PRINT   <Address: >
001166                                  PUSH    R0              ; save R0
001170  010100                          MOV     R1,R0           ; get address
001172                                  OCT16
001174                                  CRLF
001176                                  PRINT   <Expected:>
001212  010200                          MOV     R2,R0           ; get expected value
001214                                  OCT16                   ; display
001216                                  CRLF
001220                                  PRINT   <Found:    >
001234  011600                          MOV     (SP),R0         ; get bad bits
001236  074200                          XOR     R2,R0           ; generate value
001240                                  OCT16                   ; display
001242                                  CRLF                    ; space
001244                                  PRINT   <BITS:     >    ;
001260                                  POP     R0              ; get error bits
001262                                  OCT16                   ; display
001264                                  CRLF                    ; formatting
001266  000207                          RETURN                  ; done
                                ;
                                ; This routine provides error reporting for the DMA
                                ; data transfer check test. The location in memory
                                ; where bad data was found is reported.
                                ;
                                ;
001270                          DMAERR: PRINTC  <DMA data transfer error>
001322                                  PRINT   <Bad data at location >
001352  010300                          MOV     R3,R0           ; get address
001354  162700  000002                  SUB     #2,R0           ; modify
001360                                  OCT16                   ; display
001362                                  CRLF
001364  000207                          RETURN
                                ;
                                .SBTTL  Model testing information
        .NLIST  BIN
;
; This code module contains the routines necessary
; to test the following DTI interface models:
;
;
;
;
;       DT3362  A/D converter board
;
I3362   ==INIT
T3362   ==TPB
;
;
;
;
;
                                .LIST   BIN
```

```
                                        .SBTTL  Logic Tests
                                        .SBTTL  Test 1:  BRPLY from all registers
                                ;
                                ; This test verifies that the interface system responds
                                ; with a bus reply signal during a bus bus read and a
                                ; buss write cycle. All registers available on the
                                ; board are checked.
                                ;
                                ;
001366  010602                  TEST1:  MOV     SP,R2           ; save SP
001370                                  RELMOV  #3$,R0          ; set up trap to 4
001370  010700                          MOV     PC,R0           ; MOVE PC TO R0
001372  062700  000114                  ADD     #3$-.,R0        ; ADD IN RUN-TIME OFFSET
001376  010037  000004                  MOV     R0,@#4          ;
001402  013701  000542          1$:     MOV     @#BASE,R1       ; get address
001406  012704  000000                  MOV     #0,R4           ; zero register
001412                                  SCOPE                   ; declare loop point
001414  011100                          MOV     (R1),R0         ; read from board
001416  010411                          MOV     R4,(R1)         ; write to the board
001420  062701  000002                  ADD     #2,R1           ; next register
001424                                  SCOPE                   ; declare loop point
001426  011100                          MOV     (R1),R0         ; read from board
001430  010411                          MOV     R4,(R1)         ; write to the board
001432  062701  000002                  ADD     #2,R1           ; next register
001436                                  SCOPE                   ; declare loop point
001440  011100                          MOV     (R1),R0         ; read from board
001442  010411                          MOV     R4,(R1)         ; write to the board
001444  062701  000002                  ADD     #2,R1           ; next register
001450                                  SCOPE                   ; declare loop point
001452  011100                          MOV     (R1),R0         ; read from board
001454  010411                          MOV     R4,(R1)         ; write to the board
001456  062701  000002                  ADD     #2,R1           ; next register
001462                                  SCOPE                   ; declare loop point
001464  011100                          MOV     (R1),R0         ; read from board
001466  010411                          MOV     R4,(R1)         ; write to the board
001470  062701  000002                  ADD     #2,R1           ; next register
001474                                  SCOPE                   ; declare loop point
001476  011100                          MOV     (R1),R0         ; read from board
001500  010411                          MOV     R4,(R1)         ; write to the board
001502                                  SCOPE                   ; declare loop point
001504                                  EXIT                    ; leave
                                ;
                                ; *********************************************
                                ;
                                ;       Error Code 1 - Bus timeout
                                ;
                                ; *********************************************
                                ;
001506  011603                  3$:     MOV     (SP),R3         ; get offending PC
001510  010206                          MOV     R2,SP           ; restore stack
001512                                  ERROR   1,NORPLY        ; report error
001516  000113                          JMP     (R3)            ; continue test
                                ;
```

```
                                  .SBTTL  Test 2: Bit test of CSR
                          ;
                          ; This test checks all legal combinations of bits
                          ; in the CSR (Control and Status register).
                          ; Bits not checked are ERROR,DONE.
                          ;
001520  013701  000542    TEST2:  MOV     @#BASE,R1       ; get base address
001524  005004                    CLR     R4              ; init test reg.
001526  012703  100000            MOV     #32768.,R3      ; set # of states
001532                            SCOPE                   ; declare loop point
001534  010402            1$:     MOV     R4,R2           ; preload register data
001536  042702  100261            BIC     #100261,R2      ; clear all unused bits
001542  010211                    MOV     R2,(R1)         ; set bits
001544  011100                    MOV     (R1),R0         ; get bits
001546  042700  100261            BIC     #100261,R0      ; mask out floating bits
001552  074200                    XOR     R2,R0           ; test bits
001554  001402                    BEQ     3$              ; no error - skip
                          ;
                          ; **************************************************
                          ;
                          ;       Error Code 02 - bit error, RSW
                          ;
                          ; **************************************************
                          ;
001556                            ERROR   2,REG           ; report error
                          ;
001562                    3$:     SCOPE                   ; loop point
001564  062704  000002            ADD     #2,R4           ; next state
001570  077317                    SOB     R3,1$           ; loop until done
                          ;
001572                    5$:     EXIT                    ; leave
                          ;
                          ;
```

```
                                        .SBTTL  Test 3: Bit test of CFPR
                                ;
                                ; This test checks all legal combinations of bits
                                ; in the CFPR ( Channel File Program Register ).
                                ;
                                ;
001574  013701  000542          TEST3:  MOV     @#BASE,R1       ; get base address
001600  004767  177062                  CALL    PTRINT          ; initialize pointer chip
001604  005002                          CLR     R2              ; init test reg.
001606  012703  000401                  MOV     #401,R3         ; set # of states
001612                                  SCOPE                   ; declare loop point
001614  042702  177400          1$:     BIC     #177400,R2      ; clear unused bits
001620  112761  000053  000001          MOVB    #53,1(R1)       ; write pointer write command
001626  110261  000005                  MOVB    R2,5(R1)        ; program bits
001632  112761  000032  000001          MOVB    #32,1(R1)       ; prepare to read data
001640  116100  000005                  MOVB    5(R1),R0        ; read the data
001644  042700  177400                  BIC     #177400,R0      ; clear unused bits
001650  074200                          XOR     R2,R0           ; test bits
001652  001402                          BEQ     3$              ; no error - skip
                                ;
                                ; **************************************************
                                ;
                                ;       Error Code 03 - bit error, CFPR high byte
                                ;                      starting address pointer
                                ;
                                ; **************************************************
                                ;
001654                                  ERROR   3,REQ           ; report error
                                ;
001660                          3$:     SCOPE                   ; loop point
001662  005202                          INC     R2              ; next state
001664  077325                          SOB     R3,1$           ; loop until done
                                ;
                                ;
                                ;
                                ;
001666  005002                          CLR     R2               ; init test reg.
001670  012703  000401                  MOV     #401,R3          ; set # of states
001674                                  SCOPE                    ; declare loop point
001676  042702  177400          5$:     BIC     #177400,R2       ; clear unused bits
001702  112761  000063  000001          MOVB    #63,1(R1)        ; write pointer write command
001710  110261  000005                  MOVB    R2,5(R1)         ; program bits
001714  112761  000022  000001          MOVB    #22,1(R1)        ; prepare to read data
001722  116100  000005                  MOVB    5(R1),R0         ; read the data
001726  042700  177400                  BIC     #177400,R0       ; clear unused bits
001732  074200                          XOR     R2,R0            ; test bits
001734  001402                          BEQ     7$               ; no error - skip
                                ;
                                ; **************************************************
                                ;
                                ;       Error Code 04 - bit error, CFPR high byte
                                ;                       final address pointer
                                ;
                                ; **************************************************
                                ;
001736                                  ERROR   4,REG            ; report error
                                ;
001742                          7$:     SCOPE                    ; loop point
001744  005202                          INC     R2               ; next state
001746  077325                          SOB     R3,5$            ; loop until done
                                ;
                                ;
```

```
                                        ;
                                        ;
001750  112761  000053  000001                  MOVB    #53,1(R1)       ; program start address
001756  112761  000000  000005                  MOVB    #0,5(R1)        ; do it
001764  112761  000063  000001                  MOVB    #63,1(R1)       ; program end address
001772  112761  000000  000005                  MOVB    #0,5(R1)        ; do it
002000  005002                                  CLR     R2              ; init test reg.
002002  012703  000401                          MOV     #401,R3         ; set # of states
002006                                          SCOPE                   ; declare loop point
002010  042702  177400                  9$:     BIC     #177400,R2      ; clear unused bits
002014  112761  000005  000001                  MOVB    #5,1(R1)        ; write pointer write command
002022  110261  000004                          MOVB    R2,4(R1)        ; program bits
002026  112761  000004  000001                  MOVB    #4,1(R1)        ; prepare to read data
002034  116100  000004                          MOVB    4(R1),R0        ; read the data
002040  042700  177400                          BIC     #177400,R0      ; clear unused bits
002044  074200                                  XOR     R2,R0           ; test bits
002046  001402                                  BEQ     11$             ; no error - skip
                                        ;
                                        ; ************************************************
                                        ;
                                        ;       Error Code 05 - bit error, CFPR low byte,
                                        ;                       mux address data
                                        ;
                                        ; ************************************************
                                        ;
002050                                          ERROR   5,REG           ; report error
                                        ;
002054                                  11$:    SCOPE                   ; loop point
002056  005202                                  INC     R2              ; next state
002060  077325                                  SOB     R3,9$           ; loop until done
                                        ;
002062                                          EXIT                    ; leave
                                        ;
```

```
                                         .SBTTL  Test 4: Bit test of DMACSR
                                 ;
                                 ; This test checks all legal combinations of bits
                                 ; in the DMACSR ( DMA Control Status register ).
                                 ;
                                 ;
002064  013701  000542           TEST4:  MOV     @#BASE,R1          ; get base address
002070  005004                           CLR     R4                 ; init test reg.
002072  012703  000200                   MOV     #128.,R3           ; set # of states
002076                                   SCOPE                      ; declare loop point
002100  010402                   1$:     MOV     R4,R2              ; set data
002102  042702  177700                   BIC     #177700,R2         ; clear unnecessary bits
002106  010261  000006                   MOV     R2,6(R1)           ; program lower byte
002112  016100  000006                   MOV     6(R1),R0           ; read low word
002116  042700  177700                   BIC     #177700,R0         ; clear unnecessary bits
002122  074200                           XOR     R2,R0              ; test bits
002124  001402                           BEQ     3$                 ; no error - skip
                                 ;
                                 ; ************************************************
                                 ;
                                 ;       Error Code 06 - bit error, CSR
                                 ;
                                 ; ************************************************
                                 ;
002126                                   ERROR   6,REG              ; report error
                                 ;
002132                           3$:     SCOPE                      ; loop point
002134  005204                           INC     R4                 ; next state
002136  077320                           SOB     R3,1$              ; loop until done
                                 ;
002140                                   EXIT                       ; leave
                                 ;
                                 ;
```

```
                                  .SBTTL  Test 5: Bit test of DMAWCR
                          ;
                          ; This test checks all legal combinations of bits
                          ; in the DMAWCR ( DMA Word Count Register ).
                          ;
                          ;
002142  013701  000542    TEST5:  MOV     @#BASE,R1        ; get base address
002146  005004                    CLR     R4               ; init test reg.
002150  012703  000000            MOV     #0,R3            ; set # of states
002154                            SCOPE                    ; declare loop point
002156  010402            1$:     MOV     R4,R2            ; set data
002160  010261  000010            MOV     R2,10(R1)        ; program register
002164  016100  000010            MOV     10(R1),R0        ; read register
002170  074200                    XOR     R2,R0            ; test bits
002172                            SCOPE                    ; declare loop point
002174  001402                    BEQ     3$               ; no error - skip
                          ;
                          ; ***********************************************
                          ;
                          ;       Error Code 07 - bit error, DMAWCR
                          ;
                          ; ***********************************************
                          ;
002176                            ERROR   7,REG            ; report error
                          ;
002202                    3$:     SCOPE                    ; loop point
002204  005204                    INC     R4               ; next state
002206  077315                    SOB     R3,1$            ; loop until done
                          ;
002210                            EXIT                     ; leave when done
                          ;
                          ;
```

```
                                         .SBTTL  Test 6: Bit test of DMAADR
                                ;
                                ; This test checks all legal combinations of bits
                                ; in the DMAADR ( DMA Address Register ).
                                ;
                                ;
002212  013701  000542          TEST6:  MOV     @#BASE,R1       ; get base address
002216  005004                          CLR     R4              ; init test reg.
002220  012703  100002                  MOV     #32770.,R3      ; set # of states
002224                                  SCOPE                   ; declare loop point
002226  010402                  1$:     MOV     R4,R2           ; set register data
002230  010261  000012                  MOV     R2,12(R1)       ; program the register
002234  016100  000012                  MOV     12(R1),R0       ; read register
002240  074200                          XOR     R2,R0           ; test bits
002242  001402                          BEQ     3$              ; no error - skip
                                ;
                                ; ************************************************
                                ;
                                ;       Error Code 10 - bit error, DMAADR
                                ;
                                ; ************************************************
                                ;
002244                                  ERROR   10,REG          ; report error
                                ;
002250                          3$:     SCOPE                   ; loop point
002252  062704  000002                  ADD     #2,R4           ; next state
002256  077315                          SOB     R3,1$           ; loop until done
                                ;
002260                                  EXIT                    ; leave
                                ;
                                ;
```

```
                                        .SBTTL  Test 7: Check A/D START bit
                                ;
                                ; This test checks the operation of the A/D start bit.
                                ; The bit is checked for readback of zero and read-only
                                ; operation.
                                ;
002262  013701  000542          TEST7:  MOV     @#BASE,R1       ; get address
002266                                  SCOPE                   ; declare loop point
002270  005011                          CLR     (R1)            ; clear board
002272  011100                          MOV     (R1),R0         ; get bit
002274  042700  177776                  BIC     #177776,R0      ; test bit
002300  001402                          BEQ     1$              ; ok - skip error
                                ;
                                ; **************************************************
                                ;
                                ;       Error Code 11 - A/D START bit error
                                ;                       reads as a 1
                                ;
                                ; **************************************************
                                ;
002302                                  ERROR   11,REG          ; report error
002306  012711  000010          1$:     MOV     #10,(R1)        ; set external trigger bit
002312  012711  000011                  MOV     #11,(R1)        ; set start bit
002316  011100                          MOV     (R1),R0         ; get data
002320  042700  177776                  BIC     #177776,R0      ; test bit
002324  001001                          BNE     3$              ; continue no error
                                ;
                                ; **************************************************
                                ;
                                ;
                                ;       Error Code 12 - A/D START bit error
                                ;                       not read back as a 1
                                ;
                                ; **************************************************
                                ;
002326                                  ERROR   12              ; report error
                                ;
002330  005011                  3$:     CLR     (R1)            ; clear CSR
002332                                  EXIT
                                ;
```

```
                                    .SBTTL  Test 10: BINITL action
                            ;
                            ; This test verifies that the BINITL signal clears
                            ; the proper ADCSR bits. The bits that are NOT
                            ; checked are
                            ;
                            ;        mux address bits
                            ;
                            ;
002334  013701  000542      TEST10:  MOV      @#BASE,R1        ; get address
002340  004767  176340               CALL     LSETUP           ; initialize pointer
002344  005211                       INC      (R1)             ; start a conversion
002346  013700  000546               MOV      @#DELAY,R0       ; wait a while
002352  077001              1$:      SOB      R0,1$            ; do the wait
002354  005211                       INC      (R1)             ; 2nd conv to fill pipeline
002356  013700  000546               MOV      @#DELAY,R0       ; wait a while
002362  077001              3$:      SOB      R0,3$            ; do the wait
002364  005211                       INC      (R1)             ; 3rd conv to set error
002366  013700  000546               MOV      @#DELAY,R0       ; wait a while
002372  077001              5$:      SOB      R0,5$            ; do the wait
002374  012702  000116               MOV      #116,R2          ; get bits to be turned on
002400  110211                       MOVB     R2,(R1)          ; set required bits
002402  005002                       CLR      R2               ; clr test reg.
002404  013700  000546               MOV      @#DELAY,R0       ; wait a while
002410  077001              7$:      SOB      R0,7$            ; do the wait
002412                               SCOPE                     ; declare loop point
002414  000005                       RESET                     ; issue BINITL
002416  011100                       MOV      (R1),R0          ; get bits
002420  042700  000060               BIC      #60,R0           ; ignore some bits
002424  074200                       XOR      R2,R0            ; test bits
002426  001402                       BEQ      9$               ; OK - skip error
                            ;
                            ; **************************************************
                            ;
                            ;        Error Code 13 - proper bit(s) not cleared
                            ;                          by BINITL
                            ;
                            ; **************************************************
                            ;
002430                               ERROR    13,REG           ; report error
                            ;
002434                      9$:      EXIT                      ; all done
```

```
                                                .SBTTL  Test 11: Byte operation of ADCSR
                                        ;
                                        ; This test verifies high and low byte operations
                                        ; involving the ADCSR.
                                        ;
                                        ;
002436  013701  000542          TEST11: MOV     @#BASE,R1               ; get address
002442  005011                          CLR     (R1)                    ; clear ADCSR
002444  013700  000546                  MOV     @#DELAY,R0              ; wait
002450  077001                  1$:     SOB     R0,1$
002452  012702  000116                  MOV     #116,R2                 ; init. test register
002456                                  SCOPE                           ; declare loop point
002460  112711  077516                  MOVB    #77516,(R1)             ; set R/W bits
002464  011100                          MOV     (R1),R0                 ; get ADCSR as word
002466  013703  000546                  MOV     @#DELAY,R3              ; let board finish
002472  077301                  3$:     SOB     R3,3$
002474  042700  177600                  BIC     #177600,R0              ; ignore status bits
002500  074200                          XOR     R2,R0                   ; test bits
002502  001402                          BEQ     5$                      ; ok - skip error
                                        ;
                                        ; ************************************************
                                        ;
                                        ;       Error Code 14 - high byte loaded during
                                        ;                       a low byte operation
                                        ;
                                        ; ************************************************
                                        ;
002504                                  ERROR   14,REG                  ; report error
```

```
                                  ;
002510  005011                    5$:     CLR     (R1)            ; clear ADCSR
002512  013700  000546                    MOV     @#DELAY,R0      ; wait
002516  077001                    7$:     SOB     R0,7$
002520  112702  000774                    MOVB    #774,R2         ; init test register
002524                                    SCOPE                   ; declare loop point
002526  110261  000001                    MOVB    R2,1(R1)        ; set R/W bits
002532  013703  000546                    MOV     @#DELAY,R3      ; let board finish
002536  077301                    9$:     SOB     R3,9$
002540  011100                            MOV     (R1),R0         ; get ADCSR as word
002542  042700  100261                    BIC     #100261,R0      ; ignore status bits
002546  000302                            SWAB    R2              ; adjust R2
002550  042702  100377                    BIC     #100377,R2      ; clear random bits
002554  074200                            XOR     R2,R0           ; test bits
002556  001402                            BEQ     11$             ; ok - skip error
                                  ;
                                  ; ***********************************************
                                  ;
                                  ;       Error Code 15 - low byte loaded during
                                  ;                       a high byte operation
                                  ;
                                  ; ***********************************************
                                  ;
002560                                    ERROR   15,REG          ; report error
                                  ;
002564                            11$:    EXIT
```

```
                                         .SBTTL  Test 12: Test of mux/gain memory
                                 ;                         pointer chip.
                                 ;
                                 ; This test checks the operation of the mux
                                 ; memory pointer chip.
                                 ;
                                 ;
002566  013701  000542           TEST12: MOV     @#BASE,R1          ; get base address
002572  012711  000000                   MOV     #0,(R1)            ; clear CSR
002576  004767  176064                   CALL    PTRINT             ; initialize pointer chip
002602  005004                           CLR     R4                 ; init test reg.
002604  012703  000377                   MOV     #377,R3            ; set # of states
002610                                   SCOPE                      ; declare loop point
002612  004767  176204           1$:     CALL    PTRRST             ; init pointr max spread
                                 ;
002616                           3$:     SCOPE                      ; loop point
002620  010402                           MOV     R4,R2              ; load counter
002622  005202                           INC     R2                 ; bump it up
002624  042702  177400                   BIC     #177400,R2         ; clear overflow
002630  112761  000005  000001           MOVB    #5,1(R1)           ; write mux data command
002636  110261  000004           5$:     MOVB    R2,4(R1)           ; write datum
002642  112761  000032  000001           MOVB    #32,1(R1)          ; write read channel pointer
002650  116100  000005                   MOVB    5(R1),R0           ; read pointer
002654  042700  177400                   BIC     #177400,R0         ; clear some bits
002660  074200                           XOR     R2,R0              ; check the data
002662  001403                           BEQ     7$                 ; branch if no match
                                 ;
                                 ; **********************************************
                                 ;
                                 ;       Error Code 16 - No pointer chip pointer
                                 ;                       auto increment.
                                 ;
                                 ; **********************************************
                                 ;
002664                                   ERROR   16,REG             ; report error
002670  000422                           BR      9$                 ; leave
                                 ;
```

```
                                        ;
                                        ;
 002672  005204                         7$:     INC     R4              ; bump counter
 002674  005303                                 DEC     R3              ; decrement counter
 002676  001347                                 BNE     3$              ; loop
                                        ;
 002700                                         SCOPE                   ; loop point
 002702  112761  000043  000001                 MOVB    #43,1(R1)       ; write chip command
 002710  112761  000077  000005                 MOVB    #77,5(R1)       ; write strobe
 002716  112761  000032  000001                 MOVB    #32,1(R1)       ; read pointer command
 002724  116100  000005                         MOVB    5(R1),R0        ; read pointer
 002730  001402                                 BEQ     9$              ; branch if reset
                                        ;
                                        ; ************************************************
                                        ;
                                        ;       Error Code 17 - Mux pointer chip pointer
                                        ;                       not reset.
                                        ;
                                        ; ************************************************
                                        ;
 002732                                         ERROR   17              ; report error
 002734  000400                                 BR      9$              ; exit
                                        ;
 002736                                 9$:     EXIT                    ; leave
                                        ;
                                        .SBTTL  Test 13: Bit test of mux/gain memory
                                ;
                                ; This test checks all legal combinations of bits
                                ; in the channel address gain memory (   ).
                                ;
                                ;
002740  013701  000542          TEST13: MOV     @#BASE,R1       ; get base address
002744  012711  000000                  MOV     #0,(R1)         ; zero the CSR
002750  004767  175712                  CALL    PTRINT          ; initialize pointer chip
002754                                  SCOPE                   ; declare loop point
002756  004767  176040                  CALL    PTRRST          ; init pointr max spread
                                ;
002762  012704  000004                  MOV     #4,R4           ; setup memory page counter
002766                          1$:     SCOPE                   ; loop point
002770  005002                          CLR     R2              ; setup data bank
002772  012703  000400          3$:     MOV     #400,R3         ; setup counter
002776  112761  000005  000001          MOVB    #5,1(R1)        ; load memory write command
003004  110261  000004          5$:     MOVB    R2,4(R1)        ; write memory data
003010  077303                          SOB     R3,5$           ; decrement counter and loop
003012  112761  000043  000001          MOVB    #43,1(R1)       ; write pointer reset
003020  112761  000077  000005          MOVB    #77,5(R1)       ; dummy write
                                ;
003026  012703  000400                  MOV     #400,R3         ; load counter
003032  112761  000004  000001          MOVB    #4,1(R1)        ; load read memory command
                                ;
003040  116100  000004          7$:     MOVB    4(R1),R0        ; read memory data
003044  074200                          XOR     R2,R0           ; compare data
003046  001402                          BEQ     9$              ; jump if no error
                                ;
                                ; **************************************************
                                ;
                                ;       Error Code 20 - mux/gain data ram
                                ;
                                ; **************************************************
                                ;
003050                                  ERROR   20,REG          ; report error
                                ;
```

```
                                               ;
                                               ;
    003054  077307                             9$:     SOB     R3,7$           ; decrement pointer and loop
    003056  111100                                     MOVB    (R1),R0         ; get CSR low byte
    003060  062700  000002                             ADD     #2,R0           ; bump to next page
    003064  110011                                     MOVB    R0,(R1)         ; write it out
    003066  005304                                     DEC     R4              ; decrement counter
    003070  001336                                     BNE     1$              ; loop for next page
                                               ;
    003072  112711  000000                             MOVB    #0,(R1)         ; clear CSR
    003076  012703  000004                             MOV     #4,R3           ; load number of pages
    003102  005002                                     CLR     R2              ; load datum
                                               ;
    003104  112761  000043  000001             11$:    MOVB    #43,1(R1)       ; reset memory pointer
    003112  112761  000077  000005                     MOVB    #77,5(R1)       ; dummy write
    003120  012705  000377                             MOV     #377,R5         ; setup counter
    003124  112761  000005  000001                     MOVB    #5,1(R1)        ; setup write memory mode
    003132  110261  000004                     13$:    MOVB    R2,4(R1)        ; write datum
    003136  077503                                     SOB     R5,13$          ; decrement counter
    003140  111100                                     MOVB    (R1),R0         ; get CSR
    003142  062700  000002                             ADD     #2,R0           ; bump to next page
    003146  110011                                     MOVB    R0,(R1)         ; write CSR
    003150  005202                                     INC     R2              ; bump datum
    003152  077324                                     SOB     R3,11$          ; decrement counter
                                               ;
    003154  012703  000004                             MOV     #4,R3           ; load counter
    003160  012702  000000                             MOV     #0,R2           ; load datum
    003164  112761  000043  000001             15$:    MOVB    #43,1(R1)       ; load pointer reset command
    003172  112761  000077  000005                     MOVB    #77,5(R1)       ; dummy write
    003200  112761  000004  000001                     MOVB    #4,1(R1)        ; load read memory command
    003206  012705  000377                             MOV     #377,R5         ; load counter
                                               ;
```

```
                                 ;
                                 ;
003212  116100  000004           17$:    MOVB    4(R1),R0        ; read datum
003216  074200                           XOR     R2,R0           ; compare data
003220  001402                           BEQ     19$             ; check no error
                                 ;
                                 ; ***********************************************
                                 ;
                                 ;       Error Code 21 - Data- page number mismatch
                                 ;
                                 ; ***********************************************
                                 ;
003222                                   ERROR   21,REG          ; report error
                                 ;
003226  077507                   19$:    SOB     R5,17$          ; decrement counter
003230  111100                           MOVB    (R1),R0         ; get CSR
003232  062700  000002                   ADD     #2,R0           ; bump page number
003236  110011                           MOVB    R0,(R1)         ; write CSR
003240  005202                           INC     R2              ; bump datum
003242  077330                           SOB     R3,15$          ; decrement counter
                                 ;
003244  012711  000000           21$:    MOV     #0,(R1)         ; zero CSR
003250                                   EXIT                    ; leave
                                 ;
```

```
                                        .SBTTL  Test 14: A/D START bit triggering
                                ;
                                ; This test verifies that the A/D START bit of the
                                ; ADCSR triggers a conversion and that the A/D DONE
                                ; bit is cleared when data is read.
                                ;
                                ;
003252  013701  000542          TEST14: MOV     @#BASE,R1       ; get address
003256  004767  175422                  CALL    LSETUP          ; setup logic to do conversion
003262                                  DCLEAR  2$,4$           ; clear the done bit
003274                                  SCOPE                   ; declare loop point
003276  105711                          TSTB    (R1)            ; done bit clear?
003300  100002                          BPL     1$              ; yes - skip error
                                ;
                                ; ***********************************************
                                ;
                                ;       Error Code 22 - A/D DONE bit not cleared
                                ;                       after A/D data was read
                                ;
                                ; ***********************************************
                                ;
003302                                  ERROR   22              ; report error
003304  000451                          BR      17$             ; can't continue
                                ;
003306                          1$:     SCOPE                   ; declare loop point
003310  042711  100000                  BIC     #100000,(R1)    ; clear possible error bit
003314  105211                          INCB    (R1)            ; set A/D start bit
003316  013700  000546                  MOV     @#DELAY,R0      ; wait
003322  077001                  3$:     SOB     R0,3$
003324  105711                          TSTB    (R1)            ; done bit set?
003326  100402                          BMI     5$              ; yes - skip error
                                ;
                                ; ***********************************************
                                ;
                                ;       Error Code 23 - A/D DONE bit not set
                                ;
                                ; ***********************************************
                                ;
003330                                  ERROR   23              ; report error
003332  000436                          BR      17$             ; can't continue
```

```

003334                          5$:     SCOPE                   ; declare loop point
003336  016100  000002                  MOV     2(R1),R0        ; read A/D data
003342  105711                          TSTB    (R1)            ; done bit clear?
003344  100002                          BPL     7$              ; yes - skip error
                                ;
                                ; **************************************************
                                ;
                                ;       Error Code 24 - A/D DONE bit not cleared
                                ;                       after A/D data was read
                                ;
                                ; **************************************************
                                ;
003346                                  ERROR   24              ; report error
003350  000427                          BR      17$             ; can't continue
                                ;
                                ;       Board cycles compute time delay for A/D
                                ;
003352  005011                  7$:     CLR     (R1)            ; clear CSR
003354  005000                          CLR     R0              ; clear counter
003356  013702  000546                  MOV     @#DELAY,R2      ; wait to settle
003362  001022                          BNE     17$             ; leave if non zero
003364  077201                  9$:     SOB     R2,9$
003366  005211                          INC     (R1)            ; start conversion
003370  005200                  11$:    INC     R0              ; bump counter
003372  103410                          BCS     13$             ; leave if overflow
003374  105711                          TSTB    (R1)            ; wait for done
003376  100374                          BPL     11$             ; no done
003400  006300                          ASL     R0              ; multiply by 32
003402  006300                          ASL     R0              ; create new delay
003404  006300                          ASL     R0              ; create new delay
003406  006300                          ASL     R0              ; create new delay
003410  006300                          ASL     R0              ; create new delay
003412  102002                          BVC     15$             ; branch if no overflow
003414  012700  177777          13$:    MOV     #-1,R0          ; force max delay -1
003420  010037  000546          15$:    MOV     R0,@#DELAY      ; store it
003424  016100  000002                  MOV     2(R1),R0        ; clear done
003430                          17$:    EXIT
                                ;
                                .SBTTL  Test 15: Error bit operation
                        ;
                        ; This test verifies that the error bit of the ADCSR
                        ; can be set and cleared properly.
                        ;
                        ;
003432  013701  000542  TEST15: MOV     @#BASE,R1       ; get address
003436  004767  175242          CALL    LSETUP          ; init. A/D section
003442  042711  100000          BIC     #100000,(R1)    ; clear error bit
003446  013700  000546          MOV     @#DELAY,R0      ; wait
003452  077001          1$:     SOB     R0,1$
003454  005711                  TST     (R1)            ; error bit clear?
003456  100001                  BPL     3$              ; yes - skip error
                        ;
                        ; ***********************************************
                        ;
                        ;       Error Code 25 - Error bit not clear
                        ;
                        ; ***********************************************
                        ;
003460                          ERROR   25              ; report error
                        ;
003462                  3$:     SCOPE                   ; declare loop point
003464                          DCLEAR  2$,4$           ; clear done bit
003476  105211                  INCB    (R1)            ; three quick triggers
003500  013700  000546          MOV     @#DELAY,R0      ; delay for done bit
003504  077001          5$:     SOB     R0,5$           ; wait
003506  105211                  INCB    (R1)            ; second trigger
003510  013700  000546          MOV     @#DELAY,R0      ; delay for error bit
003514  077001          7$:     SOB     R0,7$
003516  105211                  INCB    (R1)            ; third trigger
003520  013700  000546          MOV     @#DELAY,R0      ; delay for error bit
003524  077001          9$:     SOB     R0,9$
003526  005711                  TST     (R1)            ; error bit set?
003530  100402                  BMI     11$             ; yes - skip error
                        ;
                        ; ***********************************************
                        ;
                        ;       Error Code 26 - Error bit not set
                        ;
                        ; ***********************************************
                        ;
003532                          ERROR   26              ; report error
003534  000473                  BR      25$
```

```

003536                              11$:    SCOPE                   ; declare loop point
003540                                      DCLEAR  6$,8$           ; clear A/D DONE bit
003552  005011                              CLR     (R1)            ; clear ADCSR
003554  013700  0C0546                      MOV     @#DELAY,R0      ; wait
003560  077001                      13$:    SOB     R0,13$
003562  005711                              TST     (R1)            ; error bit clear?
003564  100001                              BPL     15$             ; yes - skip error
                                    ;
                                    ; *************************************************
                                    ;
                                    ;       Error Code 27 - Error bit not clear
                                    ;
                                    ; *************************************************
                                    ;
003566                                      ERROR   27              ; report error
                                    ;
003570  005211                      15$:    INC     (R1)            ; start a conversion
003572  013700  000546                      MOV     @#DELAY,R0      ; wait a while
003576  077001                      17$:    SOB     R0,17$
003600  105711                              TSTB    (R1)            ; is A/D DONE set
003602  100402                              BMI     19$             ; YES - skip error
                                    ;
                                    ;**********************************************************
                                    ;
                                    ;       Error code 30 - A/D DONE bit not set
                                    ;
                                    ;**********************************************************
                                    ;
003604                                      ERROR   30
003606  000446                              BR      25$             ; cannot continue so leave
003610                              19$:    SCOPE                   ; declare loop point
003612                                      DCLEAR  10$,12$         ; clear the done
003624  005761  000002                      TST     2(R1)           ; clear A/D DONE bit for error
003630  005711                              TST     (R1)            ; check for error bit set
003632  100402                              BMI     21$             ; YES - skip error
                                    ;
                                    ; ********************************************************
                                    ;
                                    ;       Error code 31 - Error bit not set by empty
                                    ;                       pipeline when DONE bit was clear
                                    ;
                                    ;*********************************************************
                                    ;
003634                                      ERROR   31              ; report error
003636  000432                              BR      25$             ; can't continue
```

```
                                     ;
 
003640  005003                       21$:    CLR     R3
003642  012704  000340                       MOV     #PR7,R4         ; prepare for interrupt
003646  013702  000544                       MOV     @#VECTOR,R2     ; get vector address
003652                                       RELMOV  #23$,R0         ; get isr address
003652  010700                               MOV     PC,R0           ; MOVE PC TO R0
003654  062700  000040                       ADD     #23$-.,R0       ; ADD IN RUN-TIME OFFSET
003660  010062  000004                       MOV     R0,4(R2)        ; store it
003664  010462  000006                       MOV     R4,6(R2)        ; store status
003670                                       SCOPE                   ; declare loop point
003672  052711  040000                       BIS     #40000,(R1)     ; set interrupt enable
003676  106403                               MTPS    R3              ; enable cp interrupts
003700  000240                               NOP                     ; should interrupt here
003702  106404                               MTPS    R4              ; turn off cp
003704  042711  040000                       BIC     #40000,(R1)     ; clear interrupt enable
                                     ;
                                     ;   *******************************************************
                                     ;
                                     ;           Error code 32 - No interrupt on error
                                     ;
                                     ;   ******************************************************
                                     ;
003710                                       ERROR   32              ; report error
003712  000404                               BR      25$
                                     ;
003714  062706  000004               23$:    ADD     #4,SP           ; adjust the stack
003720  042711  040000                       BIC     #40000,(R1)     ; clear interrupt enable
003724                               25$:    DCLEAR  14$,16$         ; clean up
003736  005011                               CLR     (R1)
003740                                       EXIT
```

```
                                                .SBTTL  Test 16: A/D DONE interrupt
                                        ;
                                        ; This test verifies that the A/D DONE bit can
                                        ; properly generate an interrupt.
                                        ;
                                        ;
003742  013701  000542                  TEST16: MOV     @#BASE,R1       ; get address
003746  013702  000544                          MOV     @#VECTOR,R2     ; get vector address
003752  004767  174726                          CALL    LSETUP          ; setup A/D section
003756                                          DCLEAR  2$,4$           ; clear A/D DONE bit
003770                                          SCOPE                   ; loop point
003772  005011                                  CLR     (R1)            ; clear ADCSR
003774  005211                                  INC     (R1)            ; trigger a conversion
003776  013700  000546                          MOV     @#DELAY,R0      ; wait for A/D DONE
004002  077001                          1$:     SOB     R0,1$
004004  105711                                  TSTB    (R1)            ; is bit set?
004006  100402                                  BMI     3$              ; yes - skip error
                                        ;
                                        ; ************************************************
                                        ;
                                        ;       Error Code 33 - A/D DONE bit not set
                                        ;
                                        ; ************************************************
                                        ;
004010                                          ERROR   33              ; report error
004012  000427                                  BR      7$              ; can't continue
                                        ;
004014  005003                          3$:     CLR     R3              ; prepare status words
004016  012704  000340                          MOV     #PR7,R4
004022                                          RELMOV  #5$,R0          ; get ISR address
004022  010700                                  MOV     PC,R0           ; MOVE PC TO R0
004024  062700  000036                          ADD     #5$-.,R0        ; ADD IN RUN-TIME OFFSET
004030  010012                                  MOV     R0,(R2)         ; store
004032  010462  000002                          MOV     R4,2(R2)        ; store status too
```

```
                                ;
004036                                  SCOPE                     ; declare loop point
004040  152711  000100                  BISB    #100,(R1)         ; enable interrupts
004044  106403                          MTPS    R3                ; enable CPU interrupts
004046  000240                          NOP                       ; stall time
004050  106404                          MTPS    R4                ; turn off CPU
004052  142711  000100                  BICB    #100,(R1)         ; clear enable bit
                                ;
                                ; ***************************************************
                                ;
                                ;       Error Code 34 - no interrupt on A/D DONE
                                ;
                                ; ***************************************************
                                ;
004056                                  ERROR   34                ; report error
004060  000404                          BR      7$                ; can't continue
                                ;
004062  062706  000004          5$:     ADD     #4,SP             ; adjust stack
004066  142711  000100                  BICB    #100,(R1)         ; clear enable bit
                                ;
004072                          7$:     DCLEAR  6$,8$             ; clear done bit
004104                                  EXIT
                                ;
```

```
                                                 .SBTTL  Test 17: DMA logic
                                         ;
                                         ; This test verifies that the DMA logic is cycling
                                         ; properly. This test does not check the actual
                                         ; transferred data.
                                         ;
                                         ;
004106  013701  000542                   TEST17: MOV     @#BASE,R1        ; get address
004112  013702  000544                           MOV     @#VECTOR,R2      ; get vector
004116  004767  174562                           CALL    LSETUP           ; setup A/D section
004122  012703  177000                           MOV     #-1000,R3        ; get word count
004126                                           RELMOV  #DMABUF,R4       ; get buffer address
004126  010704                                   MOV     PC,R4            ; MOVE PC TO R4
004130  062704  000000C                          ADD     #DMABUF-.,R4     ; ADD IN RUN-TIME OFFSET
004134                                           DCLEAR  2$,4$            ; clear done
004146  005011                                   CLR     (R1)             ; clear board
004150  013700  000546                           MOV     @#DELAY,R0       ; wait
004154  077001                           1$:     SOB     R0,1$            ; loop
004156                                           DCLEAR  6$,8$            ; clear done
004170                                   3$:     RELMOV  #13$,R0          ; get ISR address
004170  010700                                   MOV     PC,R0            ; MOVE PC TO R0
004172  062700  000132                           ADD     #13$-.,R0        ; ADD IN RUN-TIME OFFSET
004176  010012                                   MOV     R0,(R2)          ; store
004200  012762  000340  000002                   MOV     #PR7,2(R2)       ; store status too
004206                                           SCOPE                    ; declare loop point
004210  010361  000010                           MOV     R3,10(R1)        ; load DMWCR
004214  010461  000012                           MOV     R4,12(R1)        ; load DMCAR
004220                                           DCLEAR  10$,12$          ; clear done
004232  012700  000100                           MOV     #100,R0          ; load preset
004236  010011                                   MOV     R0,(R1)          ; set INT. ENB bit
004240  112761  000005  000006                   MOVB    #5,6(R1)         ; set DMA ENB. bit
004246                                           SCOPE                    ; loop point
004250  012705  000000                           MOV     #0,R5            ; setup to enable interrupts
004254  106405                                   MTPS    R5               ; enable CPU interrupts
004256  105211                                   INCB    (R1)             ; start transfer process
                                         ;
```

```
                                        ;
004260  016105  000010                  5$:     MOV     10(R1),R5       ; sample DMWCR
004264  005705                                  TST     R5              ; end of range?
004266  001411                                  BEQ     9$              ; yes - wait for EOR
004270  013700  000546                          MOV     @#DELAY,R0      ; wait
004274  077001                          7$:     SOB     R0,7$
004276  016100  000010                          MOV     10(R1),R0       ; sample DMWCR
004302  020005                                  CMP     R0,R5           ; any transfers?
004304  001365                                  BNE     5$              ; yes - continue
                                        ;
                                        ; **************************************************
                                        ;
                                        ;       Error Code 35 - DMA failing to cycle
                                        ;
                                        ; **************************************************
                                        ;
004306                                          ERROR   35              ; report error
004310  000442                                  BR      25$             ; can't continue
                                        ;
004312  013700  000546                  9$:     MOV     @#DELAY,R0      ; wait one more cycle
004316  077001                          11$:    SOB     R0,11$
                                        ;
                                        ; **************************************************
                                        ;
                                        ;       Error Code 36 - No EOR interrupt
                                        ;
                                        ; **************************************************
                                        ;

004320                                          ERROR   36              ; report error
004322  000435                                  BR      25$             ; can't continue
```

```
                                  ;
004324  062706  000004            13$:    ADD     #4,SP               ; adjust stack
004330  005761  000010                    TST     10(R1)              ; DMAWCR=O
004334  001402                            BEQ     15$                 ; yes - skip error
                                  ;
                                  ; **************************************************
                                  ;
                                  ;       Error Code 37 - EOR interrupt without
                                  ;                       DMWCR = 0000
                                  ;
                                  ; **************************************************
                                  ;
004336                                    ERROR   37                  ; report error
004340  000426                            BR      25$                 ; can't continue
                                  ;
004342  016105  000012            15$:    MOV     12(R1),R5           ; check DMCAR
004346                                    RELMOV  #DMABUF,RO          ; get buffer address
004346  010700                            MOV     PC,RO               ; MOVE PC TO RO
004350  062700  000000C                   ADD     #DMABUF-.,RO        ; ADD IN RUN-TIME OFFSET
004354  020005                            CMP     RO,R5               ; correct?
004356  001001                            BNE     21$                 ; yes - skip error
                                  ;
                                  ; **************************************************
                                  ;
                                  ;       Error Code 40 - DMCAR failing to cycle
                                  ;
                                  ; **************************************************
                                  ;
004360                                    ERROR   40                  ; report error
```

```
                                        ;
004362                                  21$:    RELMOV  #DMABUF,R0          ; check ending address
004362  010700                                  MOV     PC,R0               ; MOVE PC TO R0
004364  062700  000000C                         ADD     #DMABUF-.,R0        ; ADD IN RUN-TIME OFFSET
004370  062700  002000                          ADD     #2000,R0            ; make number true
004374  020005                                  CMP     R0,R5               ; check it out
004376  001401                                  BEQ     23$                 ; jump
                                        ;
                                        ; *************************************************
                                        ;
                                        ;       Error Code 41 - DMA failed to stop at end
                                        ;                       of DMA transfer
                                        ;
                                        ; *************************************************
                                        ;
004400                                          ERROR   41                  ; report error
                                        ;
004402  016100  000006                  23$:    MOV     6(R1),R0            ; read DMACSR
004406  032700  000100                          BIT     #100,R0             ; error bit clear?
004412  001401                                  BEQ     25$                 ; yes - all done
                                        ;
                                        ; *************************************************
                                        ;
                                        ;       Error Code 42 - Error bit set during
                                        ;                       DMA transfer
                                        ;
                                        ; *************************************************
                                        ;
004414                                          ERROR   42                  ; report error
                                        ;
004416  012711  000000                  25$:    MOV     #0,(R1)             ; clear CSR
004422  012761  000000  000006                  MOV     #0,6(R1)            ; clear DMACSR
004430                                          EXIT                        ; all done
```

```
                                        .SBTTL  Test 20: DMA burst and semi-burst test
                                ;
                                ; This test checks the operation of DMA burst
                                ; and semi-burst modes of operation.
                                ;
                                ;
004432  013701  000542          TEST20: MOV     @#BASE,R1       ; get base address
004436  013702  000544                  MOV     @#VECTOR,R2     ; get vector address
004442                                  DCLEAR  2$,4$           ; clear the done bit
004454  004767  174224                  CALL    LSETUP          ; setup A/D section
004460  012700  000000                  MOV     #0,R0           ; setup datum
004464  112761  000053  000001          MOVB    #53,1(R1)       ; write starting address
004472  112761  000000  000005          MOVB    #0,5(R1)        ; write pointer
004500  112761  000063  000001          MOVB    #63,1(R1)       ; write ending address
004506  112761  000017  000005          MOVB    #17,5(R1)       ; write end pointer
004514  112761  000005  000001          MOVB    #5,1(R1)        ; write ram command
004522  012703  000020                  MOV     #20,R3          ; setup counter
004526  110061  000004          1$:     MOVB    R0,4(R1)        ; write address datum
004532  005200                          INC     R0              ; bump datum
004534  077304                          SOB     R3,1$           ; decrement counter and loop
                                ;
004536  012703  177000                  MOV     #-1000,R3       ; load word count
004542                                  RELMOV  #DMABUF,R4      ; load address
004542  010704                          MOV     PC,R4           ; MOVE PC TO R4
004544  062704  000000C                 ADD     #DMABUF-.,R4    ; ADD IN RUN-TIME OFFSET
004550  112761  000005  000006          MOVB    #5,6(R1)        ; program burst mode
004556  012711  000100                  MOV     #100,(R1)       ; setup CSR for interrupt
004562  010461  000012                  MOV     R4,12(R1)       ; program DMCAR
004566  010361  000010                  MOV     R3,10(R1)       ; program DMAWCR
004572                                  RELMOV  #5$,R0          ; get interrupt routine
004572  010700                          MOV     PC,R0           ; MOVE PC TO R0
004574  062700  000030                  ADD     #5$-.,R0        ; ADD IN RUN-TIME OFFSET
004600  010012                          MOV     R0,(R2)         ; store it
004602  012762  000340  000002          MOV     #PR7,2(R2)      ; program priority
004610  012700  000000                  MOV     #0,R0           ; load priority
004614  005211                          INC     (R1)            ; trigger board
004616  106400                          MTPS    R0              ; do it
                                ;
                                ;
```

```
                                          ;
                                          ;
 004620  000001                           3$:     WAIT                    ; wait for interrupt
 004622  000776                                   BR      3$              ; loop
                                          ;
 004624  062706  000004                   5$:     ADD     #4,SP           ; adjust the stack
 004630  005761  000010                           TST     10(R1)          ; check dmawcr
 004634  001402                                   BEQ     7$              ; branch if zero
                                          ;
                                          ; **************************************************
                                          ;
                                          ;       Error Code 43 - DMA interrupt DMAWCR not
                                          ;                       zero.
                                          ;
                                          ; **************************************************
                                          ;
 004636                                           ERROR   43              ; report error
 004640  000505                                   BR      27$             ; exit
                                          ;
 004642                                   7$:     DCLEAR  6$,8$           ; clear the done bit
 004654  012761  000000  000006                   MOV     #0,6(R1)        ; clear DMA done
 004662  004767  174074                           CALL    MXRST           ; call reset mux routine
 004666  012703  177000                           MOV     #-1000,R3       ; load word count
 004672                                           RELMOV  #DMABUF,R4      ; buffer pointer
 004672  010704                                   MOV     PC,R4           ; MOVE PC TO R4
 004674  062704  00000C                           ADD     #DMABUF-.,R4    ; ADD IN RUN-TIME OFFSET
 004700  010361  000010                           MOV     R3,10(R1)       ; load DMAWCR
 004704  010461  000012                           MOV     R4,12(R1)       ; load DMACAR
 004710  112761  000011  000006                   MOVB    #11,6(R1)       ; set semi-burst mode
 004716                                           RELMOV  #25$,R0         ; load address
 004716  010700                                   MOV     PC,R0           ; MOVE PC TO R0
 004720  062700  000120                           ADD     #25$-.,R0       ; ADD IN RUN-TIME OFFSET
 004724  010012                                   MOV     R0,(R2)         ; store it
 004726  012700  000340                           MOV     #PR7,R0         ; load priority
 004732  010062  000002                           MOV     R0,2(R2)        ; store it
 004736  012711  000100                           MOV     #100,(R1)       ; enable interrupts
 004742  016102  000010                           MOV     10(R1),R2       ; load pointer
 004746  012703  000037                           MOV     #37,R3          ; load counter
 004752  005000                                   CLR     R0              ; load priority
 004754  106400                                   MTPS    R0              ; enable interrupts
```

```
                                        ;
004756  005211                          13$:    INC     (R1)            ; start the board
004760  012704  000040                  15$:    MOV     #40,R4          ; load counter
004764  013700  000546                  17$:    MOV     @#DELAY,R0      ; load delay value
004770  077001                          19$:    SOB     R0,19$          ; loop
004772  077404                                  SOB     R4,17$          ; wait for channel scan
                                        ;
004774  062702  000020                          ADD     #20,R2          ; adjust the pointer
005000  016100  000010                          MOV     10(R1),R0       ; get the word count
005004  074200                                  XOR     R2,R0           ; check it out
005006  001402                                  BEQ     21$             ; loop no error
                                        ;
                                        ; *************************************************
                                        ;
                                        ;       Error Code 44 - Error in DMAWCR
                                        ;
                                        ; *************************************************
                                        ;
005010                                          ERROR   44,REG          ; report error
                                        ;
005014  005303                          21$:    DEC     R3              ; decrement the counter
005016  001357                                  BNE     13$             ; loop
                                        ;
005020  005211                                  INC     (R1)            ; trigger conversion
005022  005000                                  CLR     R0              ; set max delay
005024  077001                          23$:    SOB     R0,23$          ; wait
005026  012700  000340                          MOV     #PR7,R0         ; shut down interrupts
005032  106400                                  MTPS    R0              ; do it
                                        ;
                                        ; *************************************************
                                        ;
                                        ;       Error Code 45 - No interrupt when DMAWCR
                                        ;                       zero
                                        ;
                                        ; *************************************************
                                        ;
005034                                          ERROR   45              ; report error
005036  000406                                  BR      27$             ; leave
                                        ;
                                        ;
005040  062706  000004                  25$:    ADD     #4,SP           ; adjust the stack
005044  016100  000010                          MOV     10(R1),R0       ; get word count
005050  001401                                  BEQ     27$             ; if zero leave
                                        ;
                                        ; *************************************************
                                        ;
                                        ;       Error Code 46 - DMAWCR not zero
                                        ;
                                        ; *************************************************
                                        ;
005052                                          ERROR   46              ; report error
                                        ;
                                        ;
005054  012761  000000  000006          27$:    MOV     #0,6(R1)        ; clear DMACSR
005062  012711  000000                          MOV     #0,(R1)         ; clear CSR
005066                                          EXIT                    ; all done leave
```

```
                              .SBTTL  Test 21: DMA data transfer
                        ;
                        ; This test verifies that the DMA logic is actually
                        ; transferring data into memory.
                        ;
                        ;
005070  013701  000542  TEST21: MOV     @#BASE,R1       ; get address
005074  013702  000544          MOV     @#VECTOR,R2     ; get vector
005100  004767  173600          CALL    LSETUP          ; setup A/D section
005104                          DCLEAR  2$,4$           ; clear the done bit
005116  013700  000546          MOV     @#DELAY,R0      ; wait
005122  077001          1$:     SOB     R0,1$
005124  012703  000004          MOV     #4,R3           ; set # of iterations
005130  005005                  CLR     R5              ; clear average
005132                          SCOPE                   ; loop point
005134  105211          3$:     INCB    (R1)            ; trigger conversion
005136  013700  000546          MOV     @#DELAY,R0      ; wait for A/D DONE
005142  077001          5$:     SOB     R0,5$
005144  105711                  TSTB    (R1)            ; set?
005146  100402                  BMI     7$              ; yes - skip error
                        ;
                        ; ***********************************************
                        ;
                        ;       Error Code 4/ - A/D DONE bit not set
                        ;
                        ; ***********************************************
                        ;
005150                          ERROR   47              ; report error
005152  000530                  BR      31$             ; exit
                        ;
005154  066105  000002  7$:     ADD     2(R1),R5        ; add in data
005160  077313                  SOB     R3,3$           ; get more data
005162  006205                  ASR     R5              ; divide by 4
005164  006205                  ASR     R5              ; for average
005166  005105                  COM     R5              ; complement
                        ;***********
005170  062705  000040          ADD     #40,R5          ; force algorithm
                        ;***********
005174                          RELMOV  #DMABUF,R3      ; init. transfer area
005174  010703                  MOV     PC,R3           ; MOVE PC TO R3
005176  062703  000000C         ADD     #DMABUF-.,R3    ; ADD IN RUN-TIME OFFSET
005202  012704  002000          MOV     #2000,R4        ; number of words
005206  010523          9$:     MOV     R5,(R3)+        ; load data
005210  077402                  SOB     R4,9$           ; loop until done
```

```
  1465                                         ;
  1466 005212  012761  177400  000010                  MOV     #-400,10(R1)      ; load DMWCR
  1467 005220                                          RELMOV  #DMABUF,R0        ; get buffer address
       005220  010700                                  MOV     PC,R0             ; MOVE PC TO R0
       005222  062700  000000C                         ADD     #DMABUF-.,R0      ; ADD IN RUN-TIME OFFSET
  1468 005226  062700  000400                          ADD     #400,R0           ; add in offset
  1469 005232  010061  000012                          MOV     R0,12(R1)         ; load DMCAR
  1470 005236  112711  000100                          MOVB    #100,(R1)         ; set INT. ENB. bit
  1471 005242  112761  000005  000006                  MOVB    #5,6(R1)          ; set DMA burst enb. bit
  1472 005250                                          RELMOV  #13$,R0           ; get ISR address
       005250  010700                                  MOV     PC,R0             ; MOVE PC TO R0
       005252  062700  000032                          ADD     #13$-.,R0         ; ADD IN RUN-TIME OFFSET
  1473 005256  010012                                  MOV     R0,(R2)           ; store
  1474 005260  012762  000340  000002                  MOV     #PR7,2(R2)        ; store status too
  1475 005266                                          SCOPE                     ; loop point
  1476 005270  012700  000000                          MOV     #0,R0             ; setup to enable interrupts
  1477 005274  106400                                  MTPS    R0                ; enable CPU interrupts
  1478 005276  105211                                  INCB    (R1)              ; start transfer
  1479 005300  000001                          11$:    WAIT                      ; wait for interrupt
  1480 005302  000776                                  BR      11$               ; hang until received
  1481                                         ;
  1482                                         ; interrupt and error check sub-routine
  1483                                         ;
  1484 005304  016100  000010                  13$:    MOV     10(R1),R0         ; check dmawcr
  1485 005310  001402                                  BEQ     15$               ; ok continue on
  1486                                         ;**************************************************************
  1487                                         ;
  1488                                         ;       Error Code 50 - Error in DMAWCR
  1489                                         ;                          DMAWCR not zero
  1490                                         ;
  1491                                         ; ****************************************************
  1492                                         ;
  1493 005312                                          ERROR   50,DMAERR         ; report error
  1494                                         ;
  1495 005316  062706  000004                  15$:    ADD     #4,SP             ; adjust stack
  1496 005322                                          RELMOV  #DMABUF,R3        ; check low buffer
       005322  010703                                  MOV     PC,R3             ; MOVE PC TO R3
       005324  062703  000000C                         ADD     #DMABUF-.,R3      ; ADD IN RUN-TIME OFFSET
  1497 005330  012704  000300                          MOV     #300,R4           ; (16000 to 16600)
  1498 005334  022305                          21$:    CMP     (R3)+,R5          ; A/D data?
  1499 005336  001001                                  BNE     23$               ; yes - exit loop
  1500 005340  077403                                  SOB     R4,21$            ; no, check rest of buffer
```

```
                                  ;

005342                            23$:    RELMOV  #DMABUF,R0         ; get buffer address
005342  010700                            MOV     PC,R0              ; MOVE PC TO R0
005344  062700  000000C                   ADD     #DMABUF-.,R0       ; ADD IN RUN-TIME OFFSET
005350  062700  000402                    ADD     #402,R0            ; add in offset
005354  020003                            CMP     R0,R3              ; correct first word?
005356  001403                            BEQ     25$                ; yes - skip error
                                  ;
                                  ; **************************************************
                                  ;
                                  ;       Error Code 51 - Error in DMA transfer,
                                  ;                          starting location
                                  ;
                                  ; **************************************************
                                  ;
005360                                    ERROR   51,DMAERR          ; report error
005364  000423                            BR      31$
                                  ;
005366                            25$:    RELMOV  #DMABUF,R3         ; get buffer address
005366  010703                            MOV     PC,R3              ; MOVE PC TO R3
005370  062703  000000C                   ADD     #DMABUF-.,R3       ; ADD IN RUN-TIME OFFSET
005374  062703  000400                    ADD     #400,R3            ; add in offset
                                  ;                                  ; check rest of buffer
005400  012704  000500                    MOV     #500,R4            ; finish rest of buffer
005404  022305                    27$:    CMP     (R3)+,R5           ; A/D data?
005406  001401                            BEQ     29$                ; no - exit loop
005410  077403                            SOB     R4,27$             ; check rest of buffer
005412                            29$:    RELMOV  #DMABUF,R0         ; get buffer address
005412  010700                            MOV     PC,R0              ; MOVE PC TO R0
005414  062700  000000C                   ADD     #DMABUF-.,R0       ; ADD IN RUN-TIME OFFSET
005420  062700  001402                    ADD     #1402,R0           ; add in offset
005424  020003                            CMP     R0,R3              ; correct last word?
005426  001402                            BEQ     31$                ; yes - exit
                                  ;
                                  ; **************************************************
                                  ;
                                  ;       Error Code 52 - Error in DMA transfer,
                                  ;                          ending location
                                  ;
                                  ; **************************************************
                                  ;
005430                                    ERROR   52,DMAERR          ; report error
005434                            31$:    EXIT
```

```
                                   .SBTTL  Test 22: end of logic tests
                           ;
                           ; This test is present to inform the TST-11 monitor
                           ; that there are no more additional logic tests
                           ; to be executed. When the "ALL" command is used,
                           ; this test will force a return to the command
                           ; level of TST-11 when the test sequencer reaches
                           ; this test.
                           ;
                           ;
005436                     TEST22: ESCAPE
                           ;
                           ;
```

```
  1555                                              .SBTTL  test 23: Check DMA timeout bit operation
  1556                                      ;
  1557                                      ;  This test checks the ability of the board
  1558                                      ;  to execute a error check of a DMA transfer
  1559                                      ;  to a non-existance memoru location
  1560                                      ;
  1561 005440  013701  000542               TEST23: MOV     @#BASE,R1         ; sets the base address
  1562 005444  013702  000544                       MOV     @#VECTOR,R2       ; set vector address
  1563 005450  004767  173230                       CALL    LSETUP            ; init A/D section
  1564 005454                                       RELMOV  #3$,R0            ; sets the trap handler
       005454  010700                               MOV     PC,R0             ; MOVE PC TO R0
       005456  062700  000126                       ADD     #3$-.,R0          ; ADD IN RUN-TIME OFFSET
  1565 005462  010012                               MOV     R0,(R2)           ; set it
  1566 005464  010062  000004                       MOV     R0,4(R2)          ; set error vector
  1567 005470  012762  000340  000002               MOV     #PR7,2(R2)        ; set priority
  1568 005476  012762  000340  000006               MOV     #PR7,6(R2)        ; set priority
  1569 005504  010012                               MOV     R0,(R2)           ; set it
  1570 005506                                       DCLEAR  2$,4$             ; clear the done bit
  1571 005520  005011                               CLR     (R1)              ; clear CSR
  1572 005522  012761  160000  000012               MOV     #160000,12(R1)    ; sets DMACAR
  1573 005530  012761  177770  000010               MOV     #-10,10(R1)       ; sets # of transfers
  1574 005536  012700  000100                       MOV     #100,R0           ; interrupt enb. bit
  1575 005542  110011                               MOVB    R0,(R1)           ;
  1576 005544  112761  000065  000006               MOVB    #65,6(R1)         ; set DMA and ext. addr. bits
  1577 005552  005211                               INC     (R1)              ; starts conversion
  1578 005554                                       SCOPE                     ; loop point
  1579                                      ;
  1580 005556  005000                               CLR     R0                ; set max delay
  1581 005560  077001                       1$:     SOB     R0,1$             ; waits for interrupt
  1582 005562  012700  000000                       MOV     #0,R0             ; set to allow interrupts
  1583 005566  106400                               MTPS    R0                ; allow interrupts
  1584 005570  000240                               NOP                       ; window
  1585 005572  012700  000340                       MOV     #340,R0           ; set to stop interrupts
  1586 005576  106400                               MTPS    R0                ; shut down interrupts
  1587                                      ;
  1588                                      ;  ****************************************************************
  1589                                      ;
  1590                                      ;          Error Code 53 - No interrupt for bus
  1591                                      ;                          timout error
  1592                                      ;
  1593                                      ;  ****************************************************************
  1594                                      ;
  1595 005600                                       ERROR   53
  1596 005602  000425                               BR      9$
  1597                                      ;
```

```
                                    ;
005604  062706  000004              3$:     ADD     #4,SP           ; adjust the stack
005610                                      SCOPE                   ; loop point
005612  105761  000006                      TSTB    6(R1)           ; test dma done bit
005616  100401                              BMI     5$              ; if set continue
                                    ;
                                    ;       ************************************************************
                                    ;
                                    ;       Error Code 54 -  DMA done bit not set.
                                    ;
                                    ;       ************************************************************
                                    ;
005620                                      ERROR   54
                                    ;
005622  016100  000012              5$:     MOV     12(R1),R0       ; get word count
005626  022700  160002                      CMP     #160002,R0      ; check error address
005632  001403                              BEQ     7$              ; if zero continue
                                    ;
                                    ;       ************************************************************
                                    ;
                                    ;       Error Code 55 -  DMAACR is at incorrect value.
                                    ;                        Possible run error due to full
                                    ;                        complement of memory.
                                    ;
                                    ;       ************************************************************
                                    ;
005634                                      ERROR   55,TO
005640  000406                              BR      9$              ; leave
                                    ;
005642  016100  000006              7$:     MOV     6(R1),R0        ; read DMACSR
005646  032700  000100                      BIT     #100,R0         ; test for error bit set
005652  001001                              BNE     9$              ; yes, exit test
                                    ;
                                    ;       ************************************************************
                                    ;
                                    ;       Error code 56 -  Bus timeout error bit of DMACSR
                                    ;                        not set when non-existant memory
                                    ;                        was addressed.
                                    ;
                                    ;       ************************************************************
                                    ;
005654                                      ERROR   56              ; error
                                    ;
005656  005011                      9$:     CLR     (R1)            ; shut down board
005660  105061  000006                      CLRB    6(R1)           ; shut down DMA
005664                                      EXIT                    ; leave
                                    ;
005666                              TO:     CRLF                    ; space down
005670                                      PRINTC  < TEST WILL NOT RUN IF MEMORY >
005730                                      PRINTC  < SIZE 256 KB OR GREATER ... >
005766  000207                              RETURN
                                    ;
```

```
                                   .SBTTL  Calibration initialization
                          ;
                          ; The following routines perform initialization
                          ; functions for some of the calibration tests.
                          ;
                          ;
005770  004767  000600    GETCH:   CALL    MCINIT          ; init mux. ram
005774                             PUSH    R1              ; save pointer
005776                             RELMOV  #CHNM1,R1       ; load pointer
005776  010701                     MOV     PC,R1           ; MOVE PC TO R1
006000  062701  000474             ADD     #CHNM1-.,R1     ; ADD IN RUN-TIME OFFSET
006004  004767  000162             CALL    CHAN            ; get data
006010                             POP     R1              ; restore pointer
006012  010002                     MOV     R0,R2           ; set data
006014  005003                     CLR     R3              ; init gain
006016  105737  000540             TSTB    @#SWR           ; check for PG
006022  100002                     BPL     1$              ; no PG
006024  004767  000234             CALL    GAIN            ; get gain bits
006030  004767  000624    1$:      CALL    CHSTUP          ; set channel
006034  004767  000304             CALL    MODE            ; get mode bits
006040                             DCLEAR  2$,4$           ; clear DONE
006052  005011                     CLR     (R1)            ; clear board
006054  105061  000006             CLRB    6(R1)           ; clear DMACSR
006060  000207                     RETURN
                          ;
006062  004767  000506    GETCH1:  CALL    MCINIT          ; init mux. ram
006066                             PUSH    R1              ; save pointer
006070                             RELMOV  #CHNM2,R1       ; load pointer
006070  010701                     MOV     PC,R1           ; MOVE PC TO R1
006072  062701  000414             ADD     #CHNM2-.,R1     ; ADD IN RUN-TIME OFFSET
006076  004767  000070             CALL    CHAN            ; get data
006102  000300                     SWAB    R0              ; position
006104  010002                     MOV     R0,R2           ; save
006106                             RELMOV  #CHNM3,R1       ; load pointer
006106  010701                     MOV     PC,R1           ; MOVE PC TO R1
006110  062701  000421             ADD     #CHNM3-.,R1     ; ADD IN RUN-TIME OFFSET
006114  004767  000052             CALL    CHAN            ; get data
006120                             POP     R1              ; restore pointer
006122  074002                     XOR     R0,R2           ; add in ending channel
006124  005003                     CLR     R3              ; init gain
006126  105737  000540             TSTB    @#SWR           ; check for PG
006132  100002                     BPL     1$              ; no PG
006134  004767  000124             CALL    GAIN            ; get gain bits
006140  004767  000662    1$:      CALL    CSTUP3          ; set channel
006144  004767  000174             CALL    MODE            ; get mode bits
006150                             DCLEAR  2$,4$           ; clear DONE
006162  005011                     CLR     (R1)            ; clear board
006164  105061  000006             CLRB    6(R1)           ; clear DMACSR
006170  000207                     RETURN
                          ;
```

```
                                        ;
006172                                  CHAN:   PUSH    R1                ; save pointer
006174                                          PRINTS                    ; output prompt
006176                                          GETOCT                    ; get value
006200  103423                                  BCS     1$                ; CR at end - cont.
006202                                          PRINTC  <Enter an octal channel address>
006244                                          POP     R1                ; restore pointer
006246  000751                                  BR      CHAN              ; try again
                                        ;
006250  013700  000514                  1$:     MOV     @#ODTACC,R0       ; get value
006254  042700  177700                          BIC     #177700,R0        ; clear upper bits
006260                                          POP     R1                ; adjust the stack
006262  000207                                  RETURN                    ; return
                                        ;
                                        ;
006264                                  GAIN:   PRINT   <Gain bits? >     ; query gain bits
006302                                          GETOCT                    ; get value
006304  103412                                  BCS     1$                ; CR at end continue
006306                                          PRINTC  <Enter value 0-3>
006330  000755                                  BR      GAIN              ; loop
006332  013703  000514                  1$:     MOV     @#ODTACC,R3       ; get value
006336  042703  177774                          BIC     #177774,R3        ; clear bits
006342  000207                                  RETURN                    ; leave
                                        ;
                                        ;
006344                                  MODE:   PRINT   <Mode bits? >     ; output prompt
006362                                          GETOCT                    ; get value
006364  103420                                  BCS     1$                ; CR at end - cont.
006366                                          PRINTC  <Enter an octal byte value.>
006424  000747                                  BR      MODE              ; loop till answer correct
                                        ;
006426  013702  000514                  1$:     MOV     @#ODTACC,R2       ; get value
006432  042702  177400                          BIC     #177400,R2        ; clear high byte,R0
006436  000207                                  RETURN                    ; return
                                        ;
006440                                  ERRBT:  CRLF
006442                                          PRINT   <Error bit is set>
006466  012711  000000                          MOV     #0,(R1)           ; clear error bit
006472  000207                                  RETURN                    ; return
                                        ;
                        .NLIST  BIN
006474  CHNM1:  .ASCIZ  "Channel? "
006506  CHNM2:  .ASCIZ  "Starting channel? "
006531  CHNM3:  .ASCIZ  "Ending channel? "
006552  SMSG:   .ASCIZ  "Semi burst mode "
                .EVEN
                                                .LIST   BIN
                                        ;
                                        ;
```

```
   1747                                         ;
   1748 006574  013701  000542                  MCINIT: MOV     @#BASE,R1               ; load base address
   1749 006600  004767  172062                          CALL    PTRINT                  ; init. pointer chip
   1750 006604  112761  000053  000001                  MOVB    #53,1(R1)               ; load first address
   1751 006612  112761  000000  000005                  MOVB    #0,5(R1)                ; load beginning address
   1752 006620  112761  000063  000001                  MOVB    #63,1(R1)               ; load last address
   1753 006626  112761  000200  000005                  MOVB    #128.,5(R1)             ; load it
   1754 006634  012711  002400                          MOV     #2400,(R1)              ; write mux memory
   1755 006640  012703  000200                          MOV     #128.,R3                ; load counter
   1756 006644  005000                                  CLR     R0                      ; load channel datum
   1757 006646  110061  000004                  1$:     MOVB    R0,4(R1)                ; write channel data
   1758 006652  005200                                  INC     R0                      ; bump channel data
   1759 006654  077304                                  SOB     R3,1$                   ; decrement counter and loop
   1760 006656  000207                                  RETURN                          ; return
   1761                                         ;
   1762                                         ;
   1763 006660  010300                          CHSTUP: MOV     R3,R0                   ; position gain
   1764 006662  000300                                  SWAB    R0                      ; position bits
   1765 006664  006200                                  ASR     R0                      ;
   1766 006666  006200                                  ASR     R0                      ;
   1767 006670  150200                                  BISB    R2,R0                   ; add in channel
   1768 006672  112761  000053  000001                  MOVB    #53,1(R1)               ; load first address
   1769 006700  110261  000005                          MOVB    R2,5(R1)                ; load beginning address
   1770 006704  112761  000063  000001                  MOVB    #63,1(R1)               ; load last address
   1771 006712  110261  000005                          MOVB    R2,5(R1)                ; load it
   1772 006716  012711  002400                          MOV     #2400,(R1)              ; write mux memory
   1773 006722  110061  000004                          MOVB    R0,4(R1)                ; load gain
   1774 006726  004767  172044                          CALL    MUXLD                   ; load mux channel
   1775 006732  000207                                  RETURN                          ; return
   1776                                         ;
   1777                                         ;
   1778 006734  010300                          CSTUP2: MOV     R3,R0                   ; position gain
   1779 006736                                          PUSH    R3                      ; save data
   1780 006740  000300                                  SWAB    R0                      ; position bits
   1781 006742  006200                                  ASR     R0                      ;
   1782 006744  006200                                  ASR     R0                      ;
   1783 006746  150200                                  BISB    R2,R0                   ; add in channel
   1784 006750  112761  000053  000001                  MOVB    #53,1(R1)               ; load first address
   1785 006756  112761  000000  000005                  MOVB    #0,5(R1)                ; load beginning address
   1786 006764  112761  000063  000001                  MOVB    #63,1(R1)               ; load last address
   1787 006772  112761  000007  000005                  MOVB    #7,5(R1)                ; load it
   1788 007000  012711  002400                          MOV     #2400,(R1)              ; write mux memory
   1789 007004  012703  000010                          MOV     #10,R3                  ; load counter
   1790 007010  110061  000004                  2$:     MOVB    R0,4(R1)                ; load gain
   1791 007014  077303                                  SOB     R3,2$                   ; loop to load
   1792 007016  004767  171754                          CALL    MUXLD                   ; load mux channel
   1793 007022                                          POP     R3                      ; restore gain data
   1794 007024  000207                                  RETURN                          ; return
   1795                                         ;
```

```
                                               ;
                                               ;
 007026  010200                       CSTUP3:  MOV     R2,RO            ; get starting address
 007030  000300                                SWAB    RO               ; position it
 007032  112761  000053  000001                MOVB    #53,1(R1)        ; load first address
 007040  110061  000005                        MOVB    RO,5(R1)         ; load beginning address
 007044  010200                                MOV     R2,RO            ; position last address
 007046  112761  000063  000001                MOVB    #63,1(R1)        ; load last address
 007054  110061  000005                        MOVB    RO,5(R1)         ; load it
 007060                                        PUSH    R4               ; save data
 007062  010204                                MOV     R2,R4            ; position channels
 007064  000304                                SWAB    R4               ; position channels
 007066  042704  177700                        BIC     #177700,R4       ; clear unwanted bits
 007072  010300                                MOV     R3,RO            ; position gain
 007074  000300                                SWAB    RO               ; position bits
 007076  006200                                ASR     RO               ;
 007100  006200                                ASR     RO               ;
 007102  074400                                XOR     R4,RO            ; add in channel
 007104  012711  002400                        MOV     #2400,(R1)       ; write mux memory
 007110  110061  000004               1$:      MOVB    RO,4(R1)         ; load gain
 007114  120402                                CMPB    R4,R2            ; check for last channel
 007116  001403                                BEQ     2$               ; leave
 007120  005200                                INC     RO               ; bump next channel
 007122  005204                                INC     R4               ; bump channel address
 007124  000771                                BR      1$               ; loop
 007126  004767  171644               2$:      CALL    MUXLD            ; load mux channel
 007132                                        POP     R4               ; restore data
 007134  000207                                RETURN                   ; return
                                               ;
                                               ;
```

```
                                                       .SBTTL  Test 24: A/D Calibration
                                              ;
                                              ; This test accepts a channel address from the user
                                              ; and displays the data from that channel continuously.
                                              ;
                                              ;
007136                                        TEST24: PRINTC  <A/D Calibration>
007160  004767  176604                                CALL    GETCH           ; get channel address
007164                                                KBEXIT                  ; set up keyboard
007166  032702  000010                                BIT     #10,R2          ; external?
007172  001002                                        BNE     1$              ; yes - no start bit
007174  052702  000001                                BIS     #1,R2           ; set start bit
007200  112704  000010                        1$:     MOVB    #10,R4          ; line counter
007204                                                CRLF
007206  110211                                3$:     MOVB    R2,(R1)         ; start conversion
007210  105711                                5$:     TSTB    (R1)            ; wait for DONE
007212  100376                                        BPL     5$
007214  016100  000002                                MOV     2(R1),R0        ; get data
007220                                                OCT16                   ; display
007222  005711                                        TST     (R1)            ; error bit set?
007224  100013                                        BPL     7$              ; no - skip
007226  112700  000105                                MOVB    #'E,R0          ; print 'E'
007232                                                TTYOUT
007234  005237  000526                                INC     @#ERRCNT        ; increment count
007240  001005                                        BNE     7$              ; no overflow
007242  012737  177777  000526                        MOV     #-1,@#ERRCNT
007250  012711  000000                                MOV     #0,(R1)         ; clear error bit
007254  105304                                7$:     DECB    R4              ; line over?
007256  001750                                        BEQ     1$              ; yes - new line
007260                                                TAB
007262  000751                                        BR      3$              ; next conversion
                                              ;
```

```
                                        .SBTTL  Test 25: A/D data display under DMA
                                ;
                                ; This test allows the user to actually check the
                                ; A/D data acquired at high speed and transferred
                                ; directly into memory.
                                ;
                                ;
007264                          TEST25: PRINTC  <A/D data display under DMA>
007322  004767  176442                  CALL    GETCH              ; get channel address
007326  013703  000544                  MOV     @#VECTOR,R3        ; get vector address
007332                                  RELMOV  #9$,RO             ; get ISR address
007332  010700                          MOV     PC,RO              ; MOVE PC TO RO
007334  062700  000126                  ADD     #9$-.,RO           ; ADD IN RUN-TIME OFFSET
007340  010013                          MOV     RO,(R3)            ; store
007342                          1$:     PRINT   <Filling buffer...>
007366                                  DCLEAR  2$,4$              ; clear the done bit
007400                                  RELMOV  #DMABUF,RO         ; load DMCAR
007400  010700                          MOV     PC,RO              ; MOVE PC TO RO
007402  062700  000000C                 ADD     #DMABUF-.,RO       ; ADD IN RUN-TIME OFFSET
007406  010061  000012                  MOV     RO,12(R1)          ; load DMCAR
007412  012761  177000  000010          MOV     #-1000,10(R1)      ; load DMWCR
007420  032702  000010                  BIT     #10,R2             ; test for external trig
007424  001404                          BEQ     3$                 ; no ext. trig.
007426  112761  000011  000006          MOVB    #11,6(R1)          ; set semi burst bits
007434  000405                          BR      5$                 ; continue on
007436  052702  000001          3$:     BIS     #1,R2              ; turn on start bit
007442  112761  000005  000006          MOVB    #5,6(R1)           ; turn on dma
007450  052702  000100          5$:     BIS     #100,R2            ; set int. enb. bit
007454  110211                          MOVB    R2,(R1)            ; load mode bits
007456  000001                  7$:     WAIT                       ; wait for interrupt
007460  000776                          BR      7$                 ; hang until received
```

```
                                   ;
     007462  105011                9$:     CLRB    (R1)            ; turn off mode bits
     007464  062706  000004                ADD     #4,SP           ; adjust stack
     007470  005711                        TST     (R1)            ; error bit set?
     007472  100003                        BPL     11$             ; no - skip
     007474  004767  176740                CALL    ERRBT           ; print error bit set
     007500  000420                        BR      19$             ; loop to new setup
     007502                        11$:    RELMOV  #DMABUF,R3      ; display buffer
     007502  010703                        MOV     PC,R3           ; MOVE PC TO R3
     007504  062703  000000C               ADD     #DMABUF-.,R3    ; ADD IN RUN-TIME OFFSET
     007510  012704  001000                MOV     #1000,R4        ; conversion count
     007514  112705  000010        13$:    MOVB    #10,R5          ; line counter
     007520                                CRLF
     007522  012300                15$:    MOV     (R3)+,R0        ; get data
     007524                                OCT16                   ; display
     007526  005304                17$:    DEC     R4              ; all done?
     007530  001404                        BEQ     19$             ; yes - new buffer
     007532  105305                        DECB    R5              ; line done?
     007534  001767                        BEQ     13$             ; yes - loop
     007536                                TAB                     ; more data
     007540  000770                        BR      15$
                                   ;
     007542                        19$:    CRLF                    ; new buffer
     007544  005000                        CLR     R0              ; re-enable int.
     007546  106400                        MTPS    R0
     007550  000674                        BR      1$
                                   ;
```

```

                                       .SBTTL  Test 26: A/D input channel scan
                              ;
                              ; This test scans the input channels while
                              ; displaying the A/D data on the terminal.
                              ;
                              ;
007552                        TEST26: PRINTC  <A/D Input channel scan>
007604  004767  176764                CALL    MCINIT            ; init mux. ram
007610                                KBEXIT                    ; set up keyboard
007612  013701  000542                MOV     @#BASE,R1         ; get address
007616                                DCLEAR  2$,4$             ; clear the done bit
007630  005011                        CLR     (R1)              ; clear CSR
007632  105061  000006                CLRB    6(R1)             ; clear DMACSR
007636  005002                        CLR     R2                ; init. channel count
007640                        1$:     CRLF
007642  005003                        CLR     R3                ; init. channel gain
007644  004767  177064                CALL    CSTUP2            ; setup the channels
007650  112703  000010                MOVB    #10,R3            ; line counter
007654                                PRINT   <CH=>             ; display address
007662  010200                        MOV     R2,R0
007664                                OCT8
007666  005211                3$:     INC     (R1)              ; start the board
007670  105711                5$:     TSTB    (R1)              ; wait for DONE
007672  100376                        BPL     5$
007674  016100  000002                MOV     2(R1),R0          ; get data
007700                                TAB                       ; space over
007702                                OCT16                     ; display
007704  105303                        DECB    R3                ; line done?
007706  001367                        BNE     3$                ; no - continue
007710  005202                        INC     R2                ; inc. channel
007712  123702  000541                CMPB    @#SWR+1,R2        ; end of range?
007716  001350                        BNE     1$                ; no - continue
007720  005002                        CLR     R2                ; yes
007722                                CRLF                      ; new line
007724  000745                        BR      1$                ; loop
```

```

                                      .SBTTL  Test 27: A/D input gain/channel scan
                             ;
                             ; This test scans the input channels while
                             ; changing the gain of the converter.  The
                             ; A/D data is displaued on the terminal.
                             ;
                             ;
007726  105737  000540       TEST27: TSTB    @#SWR            ; check for gain option
007732  100113                       BPL     11$              ; gain opt. not in
007734  004767  176634               CALL    MCINIT           ; init board ram
007740                               PRINTC  <A/D Input gain/channel scan>
007776                               KBEXIT                   ; set up keyboard
010002                               DCLEAR  2$,4$            ; clear the done bit
010012  005011                       CLR     (R1)             ; clear CSR
010014  105061  000006               CLRB    6(R1)            ; clear DMACSR
010020  005002                       CLR     R2               ; init. channel count
010022  005003                       CLR     R3               ; init. gain
010024                       1$:     CRLF
010026  112704  000007               MOVB    #7,R4            ; line counter
010032                               PRINT   <CH=>            ; display address
010040  010200                       MOV     R2,R0
010042                               OCT8
010044  112700  000054               MOVB    #',,R0           ; display gain
010050                               TTYOUT
010052  112700  000060               MOVB    #'0,R0
010056  032703  000002               BIT     #2,R3            ; GS1 set?
010062  001401                       BEQ     3$               ; no - skip
010064  105200                       INCB    R0
010066                       3$:     TTYOUT                   ; display bit
010070  112700  000060               MOVB    #'0,R0
010074  032703  000001               BIT     #1,R3            ; GS0 set?
010100  001401                       BEQ     5$               ; no - skip
010102  105200                       INCB    R0
010104                       5$:     TTYOUT                   ; display bit
010106  004767  176622               CALL    CSTUP2           ; setup channel parameters
```

```
  1989                                        ;
  1990 010112  005211                         7$:      INC      (R1)              ; start the board
  1991 010114  105711                         9$:      TSTB     (R1)              ; wait for DONE
  1992 010116  100376                                  BPL      9$                ; loop
  1993 010120  016100  000002                          MOV      2(R1),R0          ; get data
  1994 010124                                          TAB                        ; space over
  1995 010126                                          OCT16                      ; display
  1996 010130  105304                                  DECB     R4                ; line done?
  1997 010132  001367                                  BNE      7$                ; no - continue
  1998 010134  005202                                  INC      R2                ; inc. channel
  1999 010136  005203                                  INC      R3                ; increment gain
  2000 010140  042703  177774                          BIC      #177774,R3        ; clear extra bits
  2001 010144  123702  000541                          CMPB     @#SWR+1,R2        ; end of range?
  2002 010150  001325                                  BNE      1$                ; no - continue
  2003 010152  005002                                  CLR      R2                ; init. channel
  2004 010154  005003                                  CLR      R3                ; init. gain
  2005 010156                                          CRLF                       ; space down
  2006 010160  000721                                  BR       1$                ; continue to loop
  2007                                        ;
  2008 010162                                 11$:     EXIT                       ; leave
  2009                                        ;
```

```
                                                .SBTTL  Test 30: A/D timing test under DMA
                                        ;
                                        ; This test allows the user to actually check the
                                        ; A/D data transfer rate at high speed.
                                        ;
                                        ;
010164                                  TEST30: PRINTC  <DMA timing test>
010206  004767  175650                          CALL    GETCH1                  ; get channel address
                                        ;
010212  005004                                  CLR     R4                      ; init dmacsr setting
010214                                          RELMOV  #SMSG,R1                ; ask about semiburst mode
010214  010701                                  MOV     PC,R1                   ; MOVE PC TO R1
010216  062701  176334                          ADD     #SMSG-.,R1              ; ADD IN RUN-TIME OFFSET
010222  004767  170156                          CALL    QUERY                   ; ask about it
010226  103403                                  BCS     1$                      ; jump if no
010230  052704  000011                          BIS     #11,R4                  ; set semi-burst mode
010234  000402                                  BR      3$                      ; jump
010236  052704  000005                  1$:     BIS     #5,R4                   ; set burst mode
                                        ;
010242  032702  000010                  3$:     BIT     #10,R2                  ; test for external trig
010246  001002                                  BNE     5$                      ; YES ext. trig.
010250  052702  000001                          BIS     #1,R2                   ; turn on start bit
010254  052702  040100                  5$:     BIS     #40100,R2               ; set all int. enb. bit
                                        ;
010260                                          KBEXIT                          ; enable way of exit
010262  013701  000542                          MOV     @#BASE,R1               ; load CSR address
010266  013703  000544                          MOV     @#VECTOR,R3             ; get vector address
010272                                          RELMOV  #11$,R0                 ; get ISR address
010272  010700                                  MOV     PC,R0                   ; MOVE PC TO R0
010274  062700  000104                          ADD     #11$-.,R0               ; ADD IN RUN-TIME OFFSET
010300  010013                                  MOV     R0,(R3)                 ; store interrupt service
010302  012763  000340  000002                  MOV     #340,2(R3)              ; store priority
010310                                          RELMOV  #15$,R0                 ; get error service routine
010310  010700                                  MOV     PC,R0                   ; MOVE PC TO R0
010312  062700  000112                          ADD     #15$-.,R0               ; ADD IN RUN-TIME OFFSET
010316  010063  000004                          MOV     R0,4(R3)                ; store it
010322  012763  000340  000006                  MOV     #340,6(R3)              ; store priority
                                        ;
010330                                  7$:     DCLEAR  2$,4$                   ; clear DONE
010342                                          RELMOV  #DMABUF,R0              ; load DMCAR
010342  010700                                  MOV     PC,R0                   ; MOVE PC TO R0
010344  062700  000000C                         ADD     #DMABUF-.,R0            ; ADD IN RUN-TIME OFFSET
010350  010061  000012                          MOV     R0,12(R1)               ; load DMCAR
010354  012761  177000  000010                  MOV     #-1000,10(R1)           ; load DMWCR
010362  110461  000006                          MOVB    R4,6(R1)                ; set DMACSR
010366  005000                                  CLR     R0                      ; set priority
010370  106400                                  MTPS    R0                      ; enable interrupts
010372  010211                                  MOV     R2,(R1)                 ; load mode bits
010374  000001                          9$:     WAIT                            ; wait for interrupt
010376  000776                                  BR      9$                      ; hang until received
                                        ;
```

```
                                   ;
                                   ; Normal interrupt routine
                                   ;
010400  105011                     11$:    CLRB    (R1)            ; turn off mode bits
010402  062706  000004                     ADD     #4,SP           ; adjust stack
010406  005711                             TST     (R1)            ; error bit set?
010410  100002                             BPL     13$             ; no - skip
010412  004767  176022                     CALL    ERRBT           ; print error bit set
010416  005000                     13$:    CLR     R0              ; re-enable int.
010420  106400                             MTPS    R0
010422  000742                             BR      7$
                                   ;
                                   ; Error interrupt service routine
                                   ;
010424  005011                     15$:    CLR     (R1)            ; turn off mode bits
010426  062706  000004                     ADD     #4,SP           ; adjust stack
010432  004767  176002                     CALL    ERRBT           ; print error bit set
010436  000734                             BR      7$
                                   ;
                                   ;
```

```
                                                .SBTTL  Test 31: A/D DMA programmable channel scan
                                        ;
                                        ; This test allows the user to actually check the
                                        ; A/D data acquired at high speed and transferred
                                        ; directly into memory.
                                        ;
                                        ;
010440                                  TEST31: PRINTC  <A/D channel scan under DMA>
010476  004767  175360                          CALL    GETCH1          ; get channel address
010502                                          PUSH    R2              ; save data
010504  112761  000043  000001                  MOVB    #43,1(R1)       ; reset pointer
010512  112761  000000  000005                  MOVB    #0,5(R1)        ; strobe
010520  112761  000032  000001                  MOVB    #32,1(R1)       ; read starting point
010526  116102  000005                          MOVB    5(R1),R2        ; read it
010532  000302                                  SWAB    R2              ; save
010534  112761  000022  000001                  MOVB    #22,1(R1)       ; read final address
010542  116100  000005                          MOVB    5(R1),R0        ; read it
010546  042700  177400                          BIC     #177400,R0      ; clear some bits
010552  074002                                  XOR     R0,R2           ; add it in
010554  112761  000004  000001                  MOVB    #4,1(R1)        ; read channel data
010562  116100  000004                          MOVB    4(R1),R0        ; get iT
010566  042700  177477                          BIC     #177477,R0      ; Strip out all but gain
010572  000302                                  SWAB    R2              ; get starting address
010574  110203                                  MOVB    R2,R3           ; position datum
010576  000302                                  SWAB    R2              ; restore the data
010600  150003                                  BISB    R0,R3           ; set gain bits
010602  004767  170214                          CALL    PTRRST          ; init pointr max spread
010606  112761  000005  000001                  MOVB    #5,1(R1)        ; write ram command
                                        ;
```