Function Block Programming for Distributed Control

A thesis presented in complete fulfilment of the requirements for the Master in Engineering

216.899 Thesis

at Massey University, Wellington
New Zealand.

Andrew Robert Meek

July 2004

I declare that this thesis is my own, unaided work. It is being submitted in complete fulfilment of the requirements for the Master of Engineering at Massey University. It has not been submitted before for any degree or examination in any other University.

....

15th day of August, 2005

MASSEY UNIVERSITY
APPLICATION FOR APPROVAL OF REQUEST TO EMBARGO A THESIS
(Pursuant to AC 98/168 (Revised 2), Approved by Academic Board 16.02.99)

Name of Candidate: Andrew Meek ID Number: 00213918

Degree: Masters Engineering Dept/Institute/School: IIS&T

Thesis Title: Function Block Programming for Distributed Control

Name of Chief Supervisor: Frans Weehuizen Telephone Extn: 6798

As author of the above named thesis, I request that my thesis be embargoed from public access

until (date) 1st July 2007 for the following reasons:

☑ Thesis contains commercially sensitive information.

□ Thesis contains information which is personal or private and/or which was given on the basis that it not be disclosed.

□ Immediate disclosure of thesis contents would not allow the author a reasonable opportunity to publish all or part of the thesis.

□ Other (specify): ______________________________

____

Please explain here why you think this request is justified:

Thesis contains commercially sensitive information and clause 6 of the project agreement signed by the student, Tait Control Systems and Massey University states: "As the project has potential commercial value, the thesis shall be embargoed for a period of 3 years." Failure to embargo this thesis would break the terms and conditions of the project agreement.

Signed (Candidate): ______________________________ Date: 24/5/04

Endorsed (Chief Supervisor): ______________________________ Date: 9-6-04

Approved/~~Not Approved~~ (Representative of VC): ______________________________ Date: 11-6-2004

*Note: Copies of this form, once approved by the representative of the Vice-Chancellor, must be bound into **every** copy of the thesis.*

11 JUN 2004

Abstract

This report discusses research and development using the draft IEC 61499 function block standard for distributed control with embedded microprocessor applications. This is a function block programming language that is currently under development for programming distributed control systems. The report covers what is required to develop an IEC 61499 compliant product and its suitablity for use with distributed control systems. To utilise the IEC 61499 standard, research and development of an embedded Java platform was performed. This required porting a Java virtual machine to run on an embedded microprocessor. An existing industrial network protocol DeviceNet was chosen for distributing the data between the network of control devices. To achieve this an upgrade was required to an existing DeviceNet communications engine to support distributed control. A third party IEC 61499 software application engine was ported to run on an embedded microprocessor. This option was chosen rather than completely developing a software engine as a commercial decision by the developer company. It also allowed support from other companies and researchers working with this standard. To test distributed control using this function block programming standard a test application consisting of a conveyor and three axis robot was developed. The test application demonstrated the feasibility of distributed control using IEC 61499 function blocks and some of the advantages of distributed control. Further outcomes of this research have highlighted some of the problems that require rectifying before this function block programming standard is feasible for commercial products.

Acknowledgments

I wish to thank TCS (Tait Control Systems Ltd), the developer company for supplying the research project, and the Foundation for Research and Technology for funding. The following staff members of the developer company are acknowledged for help and assistance throughout the research project; Peter Tait, Nathan May, Bernard Wood, Aaron Wilson, and Alistair Edgar for assistance with hardware and software development. I thank Tony Dawson of TCS for handling paper work required for external funding proposals.

I wish to thank Dr Jim Christensen of Rockwell Automation for his past research and assistance with IEC 61499 technology.

I wish to thank Dr Frans Weehuizen my supervisor at Massey University and Nathan May my supervisor at TCS.

I wish to thank Annelese Blackbourn for assisting with proof reading.

I thank the following companies and tertiary institutions for supplying hardware and software for this research.

- Sun Microsystems for their Java virtual machine.
- Rockwell Automation, for supplying their IEC 61499 run-time environment.
- Waikato Institute of Technology, for supplying the conveyor and three axis robot machinery used for the test bed application.

The following trademarks used in this report are listed below with their owners:

- DeviceNet - ODVA (Open DeviceNet Vendors Association inc).
- EtherNet/IP - ODVA.
- RSNetWorx - Rockwell Software Inc.
- IEC 61499 - International Electrotechinical Commission.

(Adapted from Meek, 2003b, p. ii).

Contents

Abstract i
Acknowledgments ii
Contents iii
List of Figures vii
List of Tables viii
Chapter 1 Introduction 1
Investigation of the Status of the IEC 61499 Specification 2
Research and Development of a Suitable Target Platform 2
Research and Development of an IEC 61499 Run-time Environment 2
Modification of Existing DeviceNet Peer-to-Peer Communication Software 3
Development of an Application 3
Chapter 2 Overview and Literature Review of Function Block Programming 4
Background to Factory Automation and Control 4
The Definition of Distributed Control 4
Industrial Programming Languages 5
Introduction to IEC 61499 Function Block Programming Language 9
Internal Operation of Function Blocks 17
Basic Function Blocks 17
Composite Function Blocks 19
Sub-application Function Blocks 20
Service Interface (SI) Function Blocks 21
Construction of a Function Block System 22
Software Implementation of the IEC 61499 Standard 24
Summary of the Programming Environments 26
Status of the IEC 61499 Standard 26
Chapter 3 Selection and Evaluation of Target Software Environment 28
Status of IEC 61499 Development at Commencement of the Project 28
Choosing a Suitable Programming Language 28
Java 2 Platform, Micro Edition 38
The Structure of Java and J2ME 39
Porting of the KVM to an Embedded Platform 41
Platform Independent Code 44
Platform Dependent Code 45
Summary 57
Chapter 4 Evaluation of Embedded Java 58
Test Procedure 58
Memory Usage 58
Processor Speed Requirements 59
Results 61
Bench Mark Testing 63

Test Set Up 63
Results 63
Conclusions 66
Chapter 5 Hardware Platform Development for Function Block Programming 69
Choice of Target Microprocessor 70
Increasing the Memory of the CPU4 platform 71
Products that Required Converting to IEC 61499 73
Upgrading the 16 Way I/O Module for Distributed Control Technology 74
Upgrading the Motor Control Station 75
Chapter 6 Selection and Development of the IEC 61499 Run-time Environment 76
Selection of an IEC 61499 Run-time Environment 76
Function Block Run-time Environment (FBRT) 78
Porting the FBRT for use with the Developer's Target Platform 79
First Test Application to get the FBRT Running on the Developer's Target Platform 80
Internal Structure of the FBRT 81
Developing Communications with the Function Block Development Kit (FBDK) 83
DeviceNet Communications 86
Master/Slave Communication Parameters 90
Building the Library of Standard Function Blocks 90
Serial Communications 91
Test Applications 92
DeviceNet Master/Slave Centralised Device Application 92
Conveyor and Robot Application 97
Evaluation of the IEC 61499 Standard at Present 114
Function Block Programming Software 114
Hardware Development 115
IEC 61499 Function Block Library 115
Chapter 7 Upgrading DeviceNet for use with the IEC61499 Standard 117
Problems with the Existing DeviceNet stack for use with IEC 61499 Run-time Environment 117
Upgrading the Application Interface and Developing the Packet Formats 120
Creating a DeviceNet Application Object 121
Peer-to-peer Communications with DeviceNet 124
Creating Explicit Messaging Connections 124
Sending Explicit Messages 125
Creating I/O Connections 126
Configuration of IEC 61499 Devices 130
Chapter 8 Conclusions 132
Project Review 132
Appendix A IEC 61499 Device Class Definitions 135
Create 136
Delete 136
Start 136
Stop 136

Kill ... 136
Query ... 136
Read ... 136
Write ... 136
Appendix B SICK Barcode Scanner Interface Technical Manual ... 137
DeviceNet Interface Technical Manual ... 138
Getting Started ... 139
DeviceNet Information ... 140
Specifications ... 149
Appendix C Raw Data From Evaluation of Java Testing ... 151
Sample Oscilloscope Screen for Measuring the Trigger Delay ... 151
Raw Data Measured over DeviceNet ... 151
Appendix D Report Sent to Developer on Developing their own Run-time Environment167
Report into Requirements for Developing an IEC 61499 Run-time Environment. 167
Abstract ... 167
Stages Required for Development an an IEC 61499 Run-time Environment ... 167
Developing a Java Platform ... 168
Development of Tools Required to Interface an IEC 61499 Run-time Environment to the PC Platform ... 171
Developing the TCS Run-time ... 173
A Calendar of Events if Developing our own Run-time Environment ... 176
Appendix E Function Block Library ... 178
Event Function Blocks ... 178
Information Exchange Function blocks ... 184
DeviceNet Specific Function Blocks ... 185
Mathematical Function Blocks ... 186
Serial Communication Function Blocks ... 193
I/O Related Function Blocks ... 193
Appendix F DeviceNet IEC 61499 Conformance Profile ... 198
1. GENERAL PROVISIONS ... 199
1.1. Scope ... 199
1.2. Normative references ... 200
1.3. Definitions ... 200
2. PORTABILITY PROVISIONS ... 201
3. INTEROPERABILITY PROVISIONS ... 201
3.1 Supported DeviceNet Connections ... 201
3.4 Presentation of Data ... 203
3.5 Application layer ... 207
4. CONFIGURABILITY PROVISIONS ... 209
4.1 Software tools ... 209
4.2 Device management services ... 209
4.3 Devices ... 210
4.4 `FBMGT` Document Type Definition (DTD) ... 211
4.5 `Request/Response` semantics ... 215

Annex A (informative) Extensions to IEC 61499 219
Annex B (informative) An example of remote device configuration 220
Annex C (informative) IEC 61499 Interface Class Specification 222
Annex D (informative) Establishment of Dynamic I/O connections with DeviceNet 222
Creation of an CLIENT/SERVER connection 223
Creation of PUBLISHER/SUBSCRIBER connections 224
Appendix G The IEC 61499 Interface Class 225
IEC 61499 Interface Object 225
Class Attributes 225
Instance Attributes 225
Common Services 227
Appendix H Wiring Schedule for Robot and Conveyor Application 228
Appendix I Object Orientation of DeviceNet and Message Formats 230
Object Orientation 230
DeviceNet Messaging Formats 231
Standard Non-fragmented Message Formats 234
Fragmented Message Formats 235
Appendix J The Connection Class 237
Definition of the Attributes 238
State Attribute 238
instance_type Attribute 239
transportClass_trigger Attribute 239
produced_connection_id Attribute 240
consumed_connection_id Attribute 240
initial_comm_charateristics Attribute 240
produced_connection_size Attribute 240
consumed_connection_size Attribute 241
expected_packet_rate Attribute 241
watchdog_timeout_action 241
produced_connection_path & produced_connection_path_length Attributes 241
consumed_connection_path & consumed_connection_path_length Attribute 242
production_inhibit_time Attribute 242
Appendix K Evaluation of Embedded Java Benchmark Software Tests 243
C Application Source Code 243
Java Application Source Code 246
Glossary 251
References 254

List of Figures

Figure 2.1 Example PLC program 5
Figure 2.2 UML state diagram 8
Figure 2.3 Resource sharing over a network 10
Figure 2.4 Declaration of a function block 11
Figure 2.5 Representation using logic gates 12
Figure 2.6 Example function block application 12
Figure 2.7 Bar code scanner function block application 14
Figure 2.8 Equivalent ladder logic program 16
Figure 2.9 Example execution control chart 18
Figure 2.10 Example of a composite function block 19
Figure 2.11 Example of distribution with sub-applications 21
Figure 2.12 Example time-sequence diagram 22
Figure 2.13 Distribution over resources 23
Figure 3.1 Basic Function Block 32
Figure 3.2 The architecture of Java 40
Figure 3.3 Simplified Diagram of the Embedded Java KVM 42
Figure 3.4 Comparison between storage structures 49
Figure 3.5 Java Debugger Interface Architecture 54
Figure 4.1 Diagram of test set up 59
Figure 4.2 Photograph of the test set up 60
Figure 5.1 CPU4 target platform before memory expansion modifications 69
Figure 5.2 CPU4 Memory Maps 72
Figure 5.3 Original bus I/O product 74
Figure 6.1 Serial loop back program 80
Figure 6.2 Internal operation of the FBRT 82
Figure 6.3 Interconnection of function blocks 83
Figure 6.4 Function block diagram for management resource 84
Figure 6.5 Proxy device used to down load function block programs into the remote device 85
Figure 6.6 PUBLISH and SUBSCRIBE Function Blocks 87
Figure 6.7 SERVER and CLIENT FUNCTION BLOCKS 87
Figure 6.8 DeviceNet Master/Slave Communications Function Block 88
Figure 6.9 DeviceNet device identification interface 89
Figure 6.10 Serial port communications interface function block 91
Figure 6.11 Function block for removing the ETX and STX characters of a serial packet92
Figure 6.12 Partial implementation of bar code scanner interface application with function blocks 93
Figure 6.13 Upgraded bar code scanner interface application 96
Figure 6.14 Photograph of test application 98
Figure 6.15 Network diagram 98
Figure 6.16 Photograph of the bar code scanner 99
Figure 6.17 Photograph of conveyor with photo eyes and rejectors 100

Figure 6.18 Photograph of three axis robot 100
Figure 6.19 Bar code scanner software 102
Figure 6.20 Sixteen way I/O module function block program 104
Figure 6.21 Conveyor sorting control function blocks 105
Figure 6.22 Position control function block 106
Figure 6.23 Robot control function blocks 108
Figure 6.24 Robot control execution control chart 109
Figure 6.25 Robot request function blocks 111
Figure 6.26 Conveyor motor controller 112
Figure 7.1 Peer-to-peer connections being created with a management node 119
Figure 7.2 Device function block packet format 122
Figure 7.3 Explicit message packet format 125
Figure 7.4 Client Connection Establishment 128
Figure 7.5 Publisher/subscriber connection establishment 129
Figure 7.6 Serial communications packet 131
Figure C.1 Sample oscilloscope readout for measuring the trigger delay 151
Figure D.1 Programming interface for IEC 61499 devices 171
Figure E.1 Information exchange function blocks 184
Figure H.1 Wiring schedule for control devices with the robot and conveyor application229
Figure I.1 RSNetWorx Class instance editor tool 232
Figure I.2 Non-fragmented explicit request format 234
Figure I.3 Non-fragmented successful response message body format 235
Figure I.4 Error response message 235
Figure I.5 I/O Message fragment format 236
Figure I.6 Explicit message fragment format 236

List of Tables

Table 3.1 File information table 50
Table 3.2 File information table used for embedded Java 51
Table 4.1 Memory usage between C and Java applications 61
Table 4.2 Response times between Java and C application 62
Table 4.3 Benchmark test results 64
Table 4.3 Benchmark without the repeat loop results 65
Table 7.1 Class identifiers 121
Table A.1 Device class definitions 135
Table C.1 The time delays of the C application 152
Table C.2 The time delays of the Java application 158
Table E.1 Event function blocks 178
Table E.2 DeviceNet communications function blocks 185
Table E.3 Mathematical operations function blocks 186
Table E.4 Serial communications function blocks 193
Table E.5 I/O related function blocks 193

Table I.1 DeviceNet service codes and names 233
Table J.1 Connection object instance attributes 238

Chapter 1 Introduction

At present a lot of factory control systems use centralised architectures with a PLC and distributed I/O. Distributed I/O is where I/O modules are remote from a central controller and distributed on a network. With a centralised control architecture there are some inherent disadvantages such as: single point of failure possible, a large central processor, proprietary programming interface, and use of significant network bandwidth. There is the view that distributed control will overcome some of these problems. Previous implementations of distributed control have had disadvantages, such as different interface standards, each device requiring individual programming, and no programming environment suitable for the skill level of those who program the control systems. There is the hope that the developing IEC 61499 standard will overcome some of the problems with both past and present technology. This is a standard that is currently under development and its use has been limited to academic applications and specification development. The aim of this research is to: Investigate the IEC 61499 specification (IEC TC65/WG6(PT1CD)4. 2003, IEC TC65/WG6(PT4PAS)FD. 2002) and build a device that is commercially acceptable to industry. The final objective was not attained, however it has highlighted the areas where further research is required. These are as follows:

- Specification evaluation: This was finding out if the IEC 61499 specification is mature enough for commercialisation.
- Research and development of a suitable hardware platform.
- Development of peer-to-peer communication. This utilised existing peer-to-peer communications software with DeviceNet, an automation field bus network to communicate with the distributed devices.
- Development of a prototype product to demonstrate IEC 61499 capabilities for commercialisation.

Investigation of the Status of the IEC 61499 Specification

The investigation of the this specification was to:

- Find out if the specification has advanced enough to develop a product using this technology.
- To gain familiarity with this standard and the programming methodology.
- Find out what other research with this technology has been performed.
- Develop a project plan showing how the research would be performed.

Research and Development of a Suitable Target Platform

The aim was to investigate and develop a target platform using function block programming. This required research in the following areas:

- What platforms other researchers have used.
- The programming language other researchers have used to develop their IEC 61499 run-time environments.
- Develop the necessary tools and platform required to complete this research.

Research and Development of an IEC 61499 Run-time Environment

An IEC 61499 run-time was researched and developed. As a result of this research, it was decided to purchase Rockwell Automation's experimental IEC 61499 run-time environment and modify it to suit the developer's target platform and applications.

Modification of Existing DeviceNet Peer-to-Peer Communication Software

Research and development of the DeviceNet peer-to-peer communication protocol has the focus of a previous research project. This is an existing industrial network protocol used in industry. For further references refer to (Open DeviceNet Vendor Association, Release 2.0 Eratta 5, vol. 1). This required modification of the developer's communication engine so it was compatible with an IEC 61499 run-time environment. The previous project added peer-to-peer communications to the developer's DeviceNet communications engine.

Development of an Application

During the process of research and development, application software had to be developed for testing the function block run-time environment, function block types and communication interfaces. The final result of the project was a demonstration of proof of concept, and as discussed in later chapters the requirements for commercialisation of this technology.

This research is being performed for a product developer Tait Control Systems Limited (TCS). Their field of work is in design and manufacture of factory automation products. These include DeviceNet communication interfaces for centralised distributed I/O control systems. Their aim is to further develop these interfaces allowing them to be end-user programmable and have distributed control functionality. The scope of this research is to investigate the feasibility and development of a prototype IEC 61499 product. The developer demonstrated a working model using this technology at the EMEX trade show at Auckland, May 2004 which created industry interest.