

# ORKID

OPEN REAL-TIME KERNEL INTERFACE DEFINITION

*Drafted by*  
The ORKID Working Group  
Software Subcommittee of VITA

Draft 2.1  
August 1990

**Copyright 1990 by VITA, the VMEbus International Trade Association**

No parts of this document may be reproduced or used in any form or any means - electronic, graphic, mechanical, electrical or chemical, photocopying, recording in any medium, taping by any computer or storage system etc without prior permission in writing from VITA, the VMEbus International Trade Association.

**Exception:**

*This document may be reproduced or multiplied by photocopying for the exclusive purpose of soliciting public comments on the draft.*

## FROM THE CHAIRMAN

Before you lies the draft of VITA's Open Real Time Interface Definition, known as ORKID. This draft is the result of the activities of a small working group under the auspices of the Software Subcommittee of the VITA Technical Committee.

The members of the working group are:

Reed Cardoza	Eyring Research	
Alfred Chao	Software Components	
Chris Eck	CERN	
Wayne Fischer	FORCE Computers	
John Fogelin	Wind River Systems	
Zoltan Hunor	VITA Europe	(secretary)
Kim Kempf	Microware	
Hugh Maaskant	Philips	(chairman)
Dick Vanderlin	Motorola	

I would like to thank these members for their efforts. Also I would like to thank the companies they represent for providing the time and expenses of these members. Without that support this draft would not have been possible.

Eindhoven January 1990

## FOREWORD

The objective of the ORKID standard is to provide a state of the art open real-time kernel interface definition that on one hand allows users to create robust and portable code, while on the other hand allowing implementors the freedom to profilate their compliant product. Borderline conditions are that the standard:

- be implementable efficiently on a wide range of microprocessors,
- imposes no unnecessary hardware or software architecture,
- be open to future developments.

Many existing kernel products have been studied to gain insight in the required functionality. As a result ORKID is, from a functional point of view, a blend of these kernels. No radical new concepts have been introduced because there would be no reasonable guarantee that these could be implemented efficiently. Also they would reduce the likelihood of acceptance in the user community. This is not to say that the functionality is meagre, on the contrary: a rich set of objects and operations has been provided.

One issue still has to be addressed: that of MMU support. Clearly, now that new microprocessors have integrated MMUs and hence the cost and performance penalties of MMU support are diminishing, it will be required in the near future. At this moment, however, it was felt that more experience is needed with MMUs in real-time environments to define a standard. It is foreseen that an addendum to this standard will address MMU support.

Furthermore it is foreseen that a companion driver interface definition will be published.

## TABLE OF CONTENTS

1. INTRODUCTION . . . . .	5
2. ORKID CONCEPTS . . . . .	6
2.1. Environment . . . . .	6
2.2. ORKID Objects . . . . .	6
2.3. Naming and Identification . . . . .	8
2.4. ORKID Operations . . . . .	8
2.5. Multi-processing . . . . .	9
2.6. ORKID conformance . . . . .	11
2.7. Layout of Operation Descriptions . . . . .	12
3. NODES . . . . .	14
3.1. NODE_IDENT . . . . .	15
3.2. NODE_FAIL . . . . .	16
3.3. NODE_INFO . . . . .	17
4. TASKS . . . . .	18
4.1. TASK_CREATE . . . . .	21
4.2. TASK_DELETE . . . . .	22
4.3. TASK_IDENT . . . . .	23
4.4. TASK_START . . . . .	24
4.5. TASK_RESTART . . . . .	25
4.6. TASK_SUSPEND . . . . .	26
4.7. TASK_RESUME . . . . .	27
4.8. TASK_SET_PRIORITY . . . . .	28
4.9. TASK_SET_MODE . . . . .	29
4.10. TASK_READ_NOTE_PAD . . . . .	30
4.11. TASK_WRITE_NOTE_PAD . . . . .	31
4.12. TASK_INFO . . . . .	32
5. REGIONS . . . . .	33
5.1. REGION_CREATE . . . . .	34
5.2. REGION_DELETE . . . . .	35
5.3. REGION_IDENT . . . . .	36
5.4. REGION_GET_SEG . . . . .	37
5.5. REGION_RET_SEG . . . . .	38
5.6. REGION_INFO . . . . .	39
6. POOLS . . . . .	40
6.1. POOL_CREATE . . . . .	41
6.2. POOL_DELETE . . . . .	42
6.3. POOL_IDENT . . . . .	43
6.4. POOL_GET_BLK . . . . .	44
6.5. POOL_RET_BLK . . . . .	45
6.6. POOL_INFO . . . . .	46
7. SEMAPHORES . . . . .	47
7.1. SEM_CREATE . . . . .	48
7.2. SEM_DELETE . . . . .	49
7.3. SEM_IDENT . . . . .	50
7.4. SEM_CLAIM . . . . .	51
7.5. SEM_RELEASE . . . . .	52
7.6. SEM_INFO . . . . .	53