

# ISO/IEC TS 24718:2025-01 (E)

## Information technology - Programming languages - Guidance for the use of the Ada Ravenscar Profile in high integrity systems

---

### Contents

	Page
Foreword.....	v
Introduction.....	vi
<b>1 Scope.....</b>	<b>1</b>
<b>2 Normative references.....</b>	<b>1</b>
<b>3 Terms and definitions.....</b>	<b>1</b>
<b>4 Motivation for the Ravenscar profile.....</b>	<b>4</b>
4.1 General.....	4
4.2 Scheduling theory.....	4
4.2.1 General.....	4
4.2.2 Tasks characteristics.....	4
4.2.3 Scheduling model.....	5
4.3 Mapping Ada to the scheduling model.....	6
4.4 Non-preemptive scheduling and Ravenscar.....	7
4.5 Other program verification techniques.....	7
4.5.1 General.....	7
4.5.2 Static analysis.....	7
4.5.3 Formal analysis.....	8
4.5.4 Formal certification.....	9
<b>5 The Ravenscar profile definition.....</b>	<b>10</b>
5.1 Background.....	10
5.2 Definition.....	10
5.3 Summary of implications of pragma Profile (Ravenscar).....	11
<b>6 Rationale.....</b>	<b>11</b>
6.1 General.....	11
6.2 Ravenscar profile restrictions.....	11
6.2.1 Static existence model.....	11
6.2.2 Static synchronization and communication model.....	13
6.2.3 Deterministic memory usage.....	14
6.2.4 Deterministic execution model.....	14
6.2.5 Simple run-time behaviour.....	16
6.2.6 Parallel semantics.....	16
6.2.7 Implicit restrictions.....	17
6.3 Ravenscar profile dynamic semantics.....	17
6.3.1 Task dispatching policy.....	17
6.3.2 Locking policy.....	17
6.3.3 Queuing policy.....	17
6.3.4 Additional run-time errors defined by the Ravenscar profile.....	18
6.3.5 Potentially-blocking operations in protected actions.....	18
6.3.6 Exceptions and the No_Exceptions restriction.....	19
6.3.7 Access to shared variables.....	19
6.3.8 Elaboration control.....	20
<b>7 Examples of use.....</b>	<b>20</b>
7.1 General.....	20
7.2 Cyclic task.....	20
7.3 Coordinated release of cyclic tasks.....	21
7.4 Cyclic tasks with precedence relations.....	22

7.5	Event-triggered tasks .....	23
7.6	Shared resource control using protected objects .....	23
7.7	Task synchronization primitives .....	24
7.8	Minimum separation between event-triggered tasks .....	25
7.9	Interrupt handlers .....	25
7.10	Catering for entries with multiple callers .....	26
7.11	Catering for protected objects with more than one entry .....	27
7.12	Programming timeouts .....	29
7.13	Further expansions to the expressive power of the Ravenscar profile .....	30
<b>8</b>	<b>Verification of Ravenscar programs .....</b>	<b>30</b>
8.1	General .....	30
8.2	Static analysis of sequential code .....	31
8.3	Static analysis of concurrent code .....	31
8.3.1	General .....	31
8.3.2	Program-wide information flow analysis .....	32
8.3.3	Absence of run-time errors .....	32
8.3.4	Elaboration errors .....	33
8.3.5	Execution errors causing exceptions .....	33
8.3.6	Max_Entry_Queue_Length and suspension object check .....	33
8.3.7	Priority ceiling violation check .....	34
8.3.8	Potentially blocking operations in a protected action .....	34
8.3.9	Task termination .....	34
8.3.10	Use of unprotected shared variables .....	35
8.4	Scheduling analysis .....	35
8.4.1	General .....	35
8.4.2	Priority assignment .....	35
8.4.3	Rate monotonic utilization-based analysis .....	36
8.4.4	Response time analysis .....	37
8.4.5	Documentation requirement on run-time overhead parameters .....	38
8.5	Formal analysis of Ravenscar programs .....	39
<b>9</b>	<b>Extended example .....</b>	<b>39</b>
9.1	General .....	39
9.2	Ravenscar application example .....	39
9.3	Code .....	41
9.3.1	General .....	41
9.3.2	Cyclic task .....	42
9.3.3	Event-response (sporadic) tasks .....	42
9.3.4	Shared resource control protected object .....	44
9.3.5	Task synchronization primitives .....	45
9.3.6	Interrupt handler .....	46
9.4	Scheduling analysis .....	47
9.5	Auxiliary code .....	48
	<b>Bibliography .....</b>	<b>52</b>