

GLOSSARY OF REAL-TIME TERMINOLOGY

- Atomic Action** Indivisible program part. In reality: smallest uninterruptible programming module used to implement →semaphores or →bolts. Unfortunately, uninterruptibility cannot be guaranteed if it is not a feature of the →Realtime Operating System.
- Bolts** A generalized semaphore used to synchronize the access to resources which allow (exclusive) writing access as well as (shared) reading access.
- Clock Device** that adds the dimension time to the otherwise only logically arranged computer programs. The clock is used to measure time intervals and/or to synchronize processes. In distributed systems, a global clock is used for these purposes. Unfortunately, global clocks can only be synchronized within certain limits, typically a few microseconds.
- Context Switching** The switching of the processor from one activity to another. Context switching occurs as different processes are scheduled for execution.
- Correctness** Correct program execution should be achieved to guarantee →deterministic behaviour which is of primary importance for real-time programs. Unfortunately, it is impossible to prove the correctness of a program. Therefore, the goal of software design must be to lower the risk to a defined level, depending on the importance of the application.
- Determinism** The action of a computer system that is completely foreseeable and reproducible, a task that can only be accomplished fully in theory.
- Embedded Systems (ES)** An ES is an electronic system embedded within a plant or an external process. The external process may comprise both a physical/chemical/technical system (usually consisting of subsystems) and also humans performing some supervising or parameter setting tasks. Essential to the external system is that it obeys its own dynamics and does not allow complete restoration of a previous state in case of failure. Coupled with this property is the requirement that the ES has to respond to the external process within prescribed time constraints (→real-time requirement).

Exception An exception is an error situation which may arise during program execution (e.g., division by zero, memory protection violation, etc). An exception may be raised in several programming languages so as to signal that the error has taken place. An exception handler is a portion of program text specifying the response to the exception.

Fault-Tolerance A method to ensure continued system operation in the presence of faults. Since it is not possible to write non-trivial programs fault-free and, since hardware components will have faults, it is crucial for the reliability of a system to be fault tolerant. It should be remembered that with the advent of Very Large Scale Integrated circuits (VLSI) and by using microprogramming techniques, the faults may already have been incorporated into the microprocessor hardware proper. Countermeasures are: introduction of redundant components both in software and hardware; ensuring graceful-degradation or exception handling.

Graceful Degradation To ensure fault-tolerance, failing system components (both in software and hardware) must be handled in such a way that the effect on the remaining system is only a partial degradation of system performance, not a total system crash.

Hard Real-Time Systems Systems designed to meet given deadlines under any circumstances. Late results are useless and tardiness may result in danger and high cost. Typical examples are fly-by-wire systems used to control all important elements of an aircraft.

Interrupt Devices may want to interrupt normal program flow intermittently, e.g., to send data. This is done by requesting an interrupt. When the interrupt is acknowledged, the processor suspends the process currently executed (-> context-switching) and calls an interrupt-handler that services the requesting device. Upon completion of the interrupt handler, normal program execution continues, i.e., the processor is switched back to the process that was interrupted.

Interrupt latency The time between the occurrence of an external interrupt and the beginning of the execution of the interrupt handler. The interrupt latency is a measure for real-time performance of a system.

Polling Cyclic checking of an external flag within a program. The program stalls until the flag is set.

Semaphore Originally a person using flags for signaling purposes. Here: a simple mechanism to ensure mutual exclusion or synchronization of separate processes. A semaphore is an integer variable on which two operations ("P" and "V") are defined. Prior to accessing a device, a process must issue a call to the procedure implementing the P operation. If the value of the semaphore is 0, the process has to wait until the value is greater than 0. If it is greater than 0, "P" decrements the value of the semaphore and the process continues to execute. After releasing the device, the process must call the procedure "V" which increments the semaphore, thereby freeing the device for access by other processes.

INDEX

A

- Accuracy of synchronizing 33
- Ada. *See* real-time programming language
- Airbus 370f
- Aircraft flight control 269
- ALGOL 8, 145
- Artificial intelligence, algorithms 399
 - expert systems 411
 - functional programming 403
 - goal-oriented programming 404
 - hard real-time environment 413
 - logic-oriented programming 403
 - neural network 408
 - object-oriented programming 404
 - programming styles 402
 - semantic network 405
 - subsymbolic approach 407
 - technical aspects 399

Automation design 94

- engineering 93
- function 94
- hardware 94
- object 94
- software 94
- technology 93

B

- BINAC 2f
- Breakdown. *See* degradation
- Bus hierarchy 262
- Buses in real-time systems 258f
 - field 271
 - parallel 257f
 - serial 259f
 - standardization 273f

C

- C 350
- C++ 214, 217f
- Cache memory 247
- Capture requirement 126
- Clock. *See* synchronization
- CSMA/CD 269, 290
- Compiler, Euclid 67f
- Computer architecture. *See also* microcomputer
 - IBM 238
 - MISP 238
 - RISC 238, 239
- Concurrency control 281f
- Coordination 36
- Correctness 21, 24
- Cost of failure 20
- Course. *See* education

D

- Database system 221f, 230, 277
- DCF77 18, 33
- Deadline, hard, soft 20
- Deadlock, prevention of 165f
- Degradation 176f, 283
- Dependability 21
- Determinism. *See* predictability
- Distributed system 277
 - application-layer services 290
 - communication 279, 284f, 286, 290
 - concurrency control 281
 - conflicts 281
 - dynamic reconfiguration of 175
 - directory structures 296
 - fault tolerance 297
 - ISO reference model 284
 - message delay 288
 - partial breakdown 283, 282
 - PEARL for 155

- PORT concept 298
- real-time applications 278
- reconfiguration 299
- response time 289
- scalable processing power 278
- time constraints 279
- virtual manufacturing device 291

Digital Flight Control System (DFCS) 381f

Digital Signal Processor (DSP) 362

- ImDSP 362

DMA 45

E

Education 421f

- curriculum 430
- laboratory work 427
- research projects 428
- schedule, semester 426f

EDVAC 2f

Embedded system 41, 123, 233

ENIAC 2f

EPOS 425

Error(s) 29

- detection 178f
- handling 178f

EFA 370

Evaluation of Ada 150f, 212f

Evaluation of PEARL 155f, 212f

Evaluation of Euclid 71f

Evaluation of ISO/OSI 286f

Events, surveillance of 163f

Exception handler 36, 163f

Expert systems 411

F

Fairness property 26

Fault-tolerance 22, 297

Field bus 270, 273

Flight control computer 281

Flight control system,

- architecture 281
- clearance for flight test 292
- control 277, 284
- equipment verification test 290
- functions 285

- measurement of input data 276
- multiprocessor structure 285
- redundancy management 275f, 280
- requirements 284
- "safe-Ada" 288
- sensor error correction 276
- software 287
- software module test 289
- task dedicated processor 282
- timing 279, 284
- validation of performance 291
- design methods 274

Fly-By-Wire (FBW) 369f

Formalisms 25

Frame, language assumptions 43

Frame superimposition 42

- overall solution algorithm 64

G

GPS 33

Graceful degradation 22

Guaranteed response time 277

H

Hard real-time condition 20

High-level languages. *See also* real-time programming language

- Ada 141, 144, 148f, 151f, 288, 293
- Algol 68 145
- Basic 141, 146
- C 142, 240
- Chill 142
- Concurrent Euclid 66
- Coral 66 141, 145
- dialects 150
- Euclid 64
- FORTRAN 141, 148f, 240
- HAL/S 144, 148
- Industrial Real-Time FORTRAN 144
- LTR 141, 148
- Modula 142, 145, 240
- Occam 146
- Pascal 66, 146, 240, 320
- PEARL 141, 144f, 149f, 153f, 156
- PL/1 148

- PL/M 142
- Real-Time Euclid 42, 64f, 76f, 144 , 148
- Real-Time Turing 67
- RTL/2 141, 145

History of the concept of time 30f

History of real-time computers 2f

- BINAC 2f, 8
- EDVAC 2f, 8
- ENIAC 2
- PDP 5f
- WHIRLWIND 3f, 7

History of real-time languages 7f

- A-0 7
- Ada 150f
- ALGOL 8
- FORTRAN 0 7f
- PEARL 152f
- Symbolic Language 7

Holography, ultrasonic 351

I

Imprecise results 176

Interrupt surveillance 163f

Interrupt system 241

Interrupt, timer 69f

Invariance property 25

ISO/OSI reference model for open systems
264, 284f

K

Knowledge base 310f

- task planning 315f

L

LAN 261, 289

Language constructs 24

Leadership in real-time technology 420

Liveness property 25

LIW computer 247

M

Mailbox 73

Mainframe 237

- cache 237
- virtual memory management 237

Manufacturing Automation Protocol. *See* MAP

MAP network 288

Manufacturing automation 277f

Manufacturing Message Specification (MMS)
290

Microcomputer system, architecture 250

- coprocessor 251
- design 250
- fault tolerance 251
- multiprocessor architecture 250
- processor bus 250
- reliability 251

Microprocessor 233, 237

- architecture 233
- cache memory 247
- chip testing 249
- design 246
- embedded system 233
- high-end general-purpose processor 233
- integration density 246
- Intel Corporation 234
- interrupt 241
- low-end 234
- MIPS 246
- non-VLSI technology 234
- performance 236
- rice 236
- RISC-type 239
- silicon technology 234, 236
- VLSI technology 236, 252, 246

Minicomputer 237

MMS 290f

Modula, *See* high-level languages

Multibus 69

multiprocessor-PEARL 155f

N

Net, colored and predicate 107

- determined 108
- fuzzy 108
- timed 108
- timed stochastic 108

Neural network 408

Nondeterministic behaviour 21, 202

O

Object-oriented programming 404
 Occam 146
 Operating system. *See* real-time operating system
 Overload handling 177f

P

Parallel processing, problems of 8, 21, 23f, 165
 Parallel bus 257f, 274
 Pascal 145
 Pdp 72f
 PEARL. *See* real-time programming language
 Petri net 93, 100
 Pipelining 247
 PLC. *See* Programmable Logic Controller
 Portability 9, 29
 PORT concept 298
 Predictability 21, 43, 143, 202
 Primitives for inter-process synchronization 43
 Process delays 56
 Processor architecture, debugging 245

- evolution 237
- floating-point support 244
- instruction set 238
- memory protection 243
- modular design 248
- monitoring 245
- multitasking support 244
- virtual memory management 242

 Processor implementation 246
 Programmable logic controller (PLC) 185

- asynchronous sequential process 203
- CAE tools for PLC 205
- critique of IEC proposal 190
- evaluation of 186f
- IEC definition 185
- IEC proposal 201
- language constructs 200
- sequential function chart 202

Q

Queue waiting 54

R

Readiness 21
 Real-time communication 54
 Real-time conditions 20f
 Real-time data processing 277
 Real-time database system 221f, 230

- access time 224
- BAPAS-DB 229
- data acquisition 222
- diskless 228
- IBM 223
- IND-SQL-AU 230
- local area network. *See* LAN 227
- locking mechanism 225
- multi-tasking access 225
- OS/2 223
- PC 222
- performance 224
- priority 225
- recovery 226
- requirements 224
- server/client configuration 221
- SQL 230
- storage 227
- transaction 225
- update 226
- VAX/VMS 222
- VAX workstation 222, 230

 Real-time features 157f
 Real-time operating system 9, 34,

- EPOS 427
- PORTOS 429

 Real-time processing, AI techniques 391f
 Real-time program segment tree 42, 46f, 51
 Real-time programming language 141

- Ada 147f
- communication 149
- deadlock 167
- desirable features 158f
- distributed system 177
- education in 422f
- Euclid 67, 147f
- error handling 181
- FORTRAN 147f
- frame 43

- HAL/S 147f
- historical development 141
- PEARL 147f, 150f, 203f
- PL/1 148f
- implementation 149
- imprecise result 178
- LTR 147f
- monadic operator 164
- parallel processing 148, 166
- PL/1 147f
- process control 145, 148
- resource claiming 44
- requirements 142
- shortcomings of 145f
- schedulability 43
- software verification 182
- static language 184
- task scheduling 149
- task synchronisazion 148
- transient overload 179
- survey 147

Real-time system 123

- definition 18f
- modeling control 128

Real-time systems curriculum 419. *See* education

- course materials 428
- educational program 421f
- graduate research 428
- laboratory work 427
- PORTOS 429

Register model, frame cache 240

- RISC 240
- stack cache 240

Relativity, theory of 13f

Reliability requirements 24f

Rendezvous 27, 201, 205f, 215f

Research projects 428

Response-time, guaranteed 54

Right to speak, master process 267

- time division multiplex access 268
- token passing 268

RISC 7, 238, 239. *See also* microprocessor

Robot Belgrade USC hand 344f

Robot, control 345

- CGR search 323f, 326f
- data acquisition 357f
- data processing 362
- goal-directed search 324f
- hand 346f
- image generation 356f
- image processing 358f
- interfaces 312
- LSA 334
- mechanics 345
- motion planning 322
- multi-fingered dextrous hand 343
- object representation 327
- position controller 350
- programming 306f
- PUMA 337
- sensors 331, 349f
- subgoal generation 325
- task planner 316f
- ultrasonic sensing 351
- ultrasonic transducer 353

Robots, actors 333

- hardware structure of execution layer 337
- knowledge-based task planning 315
- LSA net 335
- modeling concepts of knowledge base 310f
- motion planning 322
- programming 305
- robot-level programming 306
- sensor integration 331f
- task-level programming 307
- teleoperation 306

Robustness 21

RSX11-M 5

RT-11 5

Run-time estimation 169f

Run-time simulation 56

S

SA/RT for real-time systems, CASE tools 136

- context diagram 133
- control flow diagram 129f
- data flow diagram 133f
- ProMod tools 138
- state transition diagram 129, 135

Safety property 25

Schedulability 34

Schedulability analysis, applications 72f

- back end 51f
- converting process tree 51
- evaluation 71
- frame superimposition 59f
- front end 45f
- front end statistics 50
- hardware assumptions 44
- implementation 66
- language 64
- language assumptions 44
- language-dependent front end 45
- language-independent back end 45
- real-time model 54
- segment 46
- segment tree 42, 46f, 51
- software assumptions 44

Schedulability analyzer 66, 68f

- evaluation summary 80
- hardware 69
- program 45

Scheduling algorithm 166f

Semaphore 27, 201, 204f, 292f

Sensor 331f, 343f

Serial bus 259

- access procedure 265f
- standardization 274
- time behavior 265

Signal processor 362f

Simultaneity 13, 21

Soft real-time condition 20

Software verification 180

Special purpose processor 246

Structured Analysis (SA/RT) 126f, 129

Superpipelined computer 247

Superscalar computer 247

Synchronization 160, 202f

- clock(s) 33, 202, 280
- concept(s) 26, 35f, 203f, 212
- conflicts 36, 202f
- languages 213
- rendezvous 27, 201, 205
- semaphore 27, 201, 204
- time slot based 35f

T

Task oriented virtual storage management 176

Teach-in 305

Teleoperation 306

Temporal order 14f

Time concept 12f, 28, 30f, 33

- frame 221
- philosophical aspects of
- regulation 34
- services 16
- slot based synchronization 35
- structure 14
- suppression 28
- unit of 17

Timeout 22, 28f

Time services 16f

Timeliness 19f

Timing, exact 166

Timing properties, quantitative 26

Token-passing 268. *See* right to speak

Tools, CASE 136

Tracing 180

Transient overloads 177f

U

Ultrasonic sensor 343

Universal time (UT) 16f

UNIX 9, 68, 214

UNIVAC 2

UTC 18

V

VAX 9, , 222, 230, 328f

VAXELN 338

Verification of software 180

Virtual manufacturing devices 291

Virtual memory, problems of 21

Virtual storage management 174

Von Neumann's architecture 8, 23

VLIW computer 247

W

WHIRLWIND

Workstation 230

Worst-case delay bounds 62f