
References

1. Beyer, S., Jacobi, C., Kroning, D., Leinenbach, D., Paul, W.: Instantiating uninterpreted functional units and memory system: Functional verification of the VAMP. In: Geist, D., Tronci, E. (eds.) CHARME 2003. LNCS, vol. 2860, pp. 51–65. Springer, Heidelberg (2003)
2. Cohen, E., Paul, W., Schmaltz, S.: Theory of multi core hypervisor verification. In: Emde Boas, P., Groen, F.C.A., Italiano, G.F., Nawrocki, J., Sack, H. (eds.) SOFSEM 2013: Theory and Practice of Computer Science. LNCS, vol. 7741, pp. 1–27. Springer, Heidelberg (2013)
3. Dalinger, I., Hillebrand, M.A., Paul, W.J.: On the verification of memory management mechanisms. In: Borriane, D., Paul, W. (eds.) CHARME 2005. LNCS, vol. 3725, pp. 301–316. Springer, Heidelberg (2005)
4. Emerson, E.A., Kahlon, V.: Rapid parameterized model checking of snoopy cache coherence protocols. In: Garavel, H., Hatcliff, J. (eds.) TACAS 2003. LNCS, vol. 2619, pp. 144–159. Springer, Heidelberg (2003)
5. Keller, J., Paul, W.J.: Hardware Design. Teubner-Texte zur Informatik. Teubner, Stuttgart (1995)
6. Kröning, D.: Formal Verification of Pipelined Microprocessors. PhD thesis, Saarland University (2001)
7. Lamport, L.: How to make a multiprocessor computer that correctly executes multiprocess programs. *IEEE Trans. Comput.* 28(9), 690–691 (1979)
8. Maisuradze, G.: Implementing and debugging a pipelined multi-core MIPS machine. Master’s thesis, Saarland University (2014)
9. MIPS Technologies, Inc. MIPS32 Architecture For Programmers – Volume 2 (March 2001)
10. Müller, C., Paul, W.: Complete formal hardware verification of interfaces for a FlexRay-like bus. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV 2011. LNCS, vol. 6806, Springer, Heidelberg (2011)
11. Müller, S.M., Paul, W.J.: On the correctness of hardware scheduling mechanisms for out-of-order execution. *Journal of Circuits, Systems, and Computers* 8(02), 301–314 (1998)
12. Müller, S.M., Paul, W.J.: Computer Architecture, Complexity and Correctness. Springer, Heidelberg (2000)
13. Pong, F., Dubois, M.: A survey of verification techniques for cache coherence protocols (1996)

14. Schmaltz, J.: A formal model of clock domain crossing and automated verification of time-triggered hardware. In: FMCAD, pp. 223–230. IEEE Computer Society, Los Alamitos (2007)
15. Schmaltz, S.: Towards the Pervasive Formal Verification of Multi-Core Operating Systems and Hypervisors Implemented in C. PhD thesis, Saarland University, Saarbrücken (2013)
16. Sweazey, P., Smith, A.J.: A class of compatible cache consistency protocols and their support by the IEEE futurebus. SIGARCH Computer Architecture News 14(2), 414–423 (1986)
17. Weaver, D.L.: OpenSPARC internals. Sun Microsystems (2008)

Index

- abstract cache 210
 - cache coherence 211
 - cache hit 210
 - cache line states 210
 - configuration 210
 - implemented memory abstraction 210
 - shared memory abstraction 211
- adder 99
- ALU *see* arithmetic logic unit
- arithmetic logic unit 106, 150
- arithmetic unit 101
 - data paths 103, 107
 - negative bit 104
 - overflow bit 106
- atomic MOESI protocol *see* MOESI protocol
- AU *see* arithmetic unit
- BCE *see* branch condition evaluation unit
- binary numbers 15
 - addition 17
 - decomposition 16
 - subtraction algorithm 20
- bit-strings 11
 - as binary numbers 15
 - as two's complement numbers 18
 - bit-operations 12
 - bytes 11
- bits 11
- Boolean equations 22
 - identities 22, 23
 - solving equations 23, 25
- Boolean expressions 20
 - evaluation 21
 - translating into circuits 33
- Boolean values 8
- branch condition evaluation unit 113, 147
- bytes 11
 - concatenation 129
- cache *see* abstract cache
- cache coherence 211
- cache consistency *see* cache coherence
- cache hit
 - abstract cache 210
 - direct mapped cache 213
 - fully associative cache 217
 - k -way associative cache 215
- CAS
 - MIPS ISA 312
 - sequential processor 313
- compare-and-swap *see* CAS
- congruence mod *see* equivalence mod
- control automata
 - shared memory system 247
 - transitions and control signals 249
- control automaton 75
 - Mealy automaton 76
 - implementation 80
 - precomputing outputs 81
- Moore automaton 76
 - implementation 76
 - precomputing outputs 78
- cyclic left shifter 110
- cyclic right-left shifter 110

- decision cycle of an access 282
- decoder 37
- decomposition lemma 16
- delayed PC 162
- detailed hardware model 44
 - circuit semantics 48
 - parameters 44
 - register semantics 46
 - reset 46
 - simulation by digital model 50, 61
 - stable signal 45
 - timing analysis 49
- digital circuit 30, 41
 - cycle 32
 - hardware computation 42
 - hardware configuration 42
 - path 32
- digital gates 30
 - n -bit gates 35
- direct mapped cache 212
 - abstraction 213
 - cache hit 213
- disjunctive normal form 26
 - complete 27

- effective address 130, 153
- end cycle of an access 281
- equivalence mod 12
 - equivalence relation 12
 - properties 13
 - solutions 14
 - system of representatives 13

- finite state transducer *see* control automaton
- forwarding *see* pipelined processor
- full adder 33
- full bits 170
- fully associative cache 216
 - abstraction 218
 - cache hit 217

- general purpose register file *see* GPR
- geometric sums 14
- glitch 46
- GPR 120, 134, 147, 159

- half adder 33
- half-decoder 38

- hazard signals
 - multi-core pipelined processor 327
 - single-core pipelined processor 197

- implementation registers *see* visible registers
- incrementer 100
- instruction set architecture *see* MIPS ISA
- integers 8
 - two's complement representation 18
- inverter 34
- invisible registers 161
 - multi-core pipelined processor 327
 - single-core pipelined processor 175
- ISA *see* MIPS ISA

- k -way associative cache 214
 - abstraction 216
 - cache hit 215

- liveness
 - multi-core pipelined processor 341
 - shared memory system 310
 - single-core pipelined processor 205

- main memory 69
 - clean operation 72
 - operating conditions 70
 - stable inputs 70
 - timing 69
- memory accesses 220
 - multi-port access sequences 221
 - processor data accesses 318, 325
 - processor instruction fetch accesses 319, 325
 - sequential access sequences 221
- memory embedding 135
- memory slices *see* memory systems
- memory system, shared *see* shared memory system
- memory systems 219
 - abstraction 220
 - cache abstraction 223
 - hardware configurations 223
 - memory accesses *see* memory accesses
 - memory slices 220
 - sequential consistency 222, 308

- memory, user-visible 219
 - memory accesses *see* memory accesses
 - sequential semantics 221
- MIPS ISA 117
 - ALU-operations 124
 - branches and jumps 127
 - J-type jumps 128
 - jump and link 128
 - R-type jumps 127
 - CAS 312
 - configuration 120
 - computation 120
 - next configuration 120
 - delayed PC 162
 - instructions
 - current instruction 121
 - decoding 123
 - I-type 118, 121
 - immediate constant 122
 - instruction fields 122
 - J-type 119, 121
 - opcode 121
 - R-type 119, 121
 - loads and stores 130
 - effective address 130
 - loads 131
 - stores 130
 - memory 129
 - multi-core *see* multi-core MIPS ISA
 - shift unit operations 126
 - summary 132
- mod operator 14
- MOESI protocol 210, 224
 - algebraic specification 230
 - invariants 224
 - master transitions 226
 - properties 234
 - slave transitions 226
- multi-core MIPS ISA 317
 - computation 317
 - configuration 317
- multi-core pipelined processor 326
 - correctness 337
 - ends of memory accesses 331
 - hazard signals 327
 - invisible registers 327
 - liveness 341
 - memory inputs stability 330
 - memory interfaces 329
 - scheduling functions 334
 - stepping function 334
- multi-core sequential processor 318
 - configuration 318
 - correctness 321
 - data accesses 318, 325
 - instruction count 323
 - instruction fetch accesses 319, 325
 - local step numbers 323
 - simulation relation 321
- multiplexer 34
- natural numbers 8
 - binary representation 15
- number of caches 219
- number of processors 219
- open collector bus 55
- open collector driver 55
- overlapping accesses 288
- parallel prefix 39
- PC 120, 134
- pipelined processor 167
 - correctness proof 178
 - proof obligations 179
 - correctness statement 177
 - forwarding 190
 - correctness proof 193
 - software conditions 191
 - full bits 170
 - invisible registers 175
 - liveness 205
 - multi-core *see* multi-core pipelined processor
 - scheduling functions 172
 - properties 173
 - with forwarding 192
 - with stalling 198
 - software conditions 176
 - stall engine 171, 196
 - correctness 203
 - hazard signals 177
 - update enable 171
 - used registers 175
 - visible registers 168
- program counter *see* PC

- RAM 83
 - 2-port CS RAM 97
 - 2-port RAM 94
 - cache state RAM 89
 - GPR RAM 92
 - multi-bank RAM 86
 - RAM-ROM 95
 - SPR RAM 90
 - SRAM 83
- random access memory *see* RAM
- read only memory *see* ROM
- registers 54
- relation 12
 - equivalence 12
 - reflexive 12
 - symmetric 12
 - transitive 12
- ROM 85
 - RAM-ROM 95
- scheduling functions
 - multi-core pipelined processor 334
 - single-core pipelined processor 172
 - properties 173
 - with forwarding 192
- self destructing hardware 61
- sequences 9
 - concatenation 11
 - subsequences 11
- sequential consistency 222, 308
- sequential processor 133
 - ALU environment 150
 - CAS 313
 - computation 134
 - configuration 134
 - correctness 139
 - delayed PC 163
 - effective address 153
 - initialization 141
 - instruction decoder 143
 - instruction fetch 142
 - jump and link 152
 - memory stage 156
 - multi-core *see* multi-core sequential processor
 - next PC environment 147
 - reading from GPR 147
 - shift for load 158
 - shift for store 154
 - shift unit environment 151
 - simulation relation 138
 - software conditions 133
 - stages of instruction execution 140, 165
 - visible registers 167
 - writing to GPR 159
- set-clear flip-flops 64
- shared memory system 235
 - cache abstraction 223
 - control automata 247
 - transitions and control signals 249
 - correctness proof 261
 - 1 step 301
 - accesses of the hardware computation 279
 - arbitration 261
 - automata synchronization 264
 - auxiliary registers 267
 - classification of accesses 280
 - control of tristate drivers 269
 - data transmission 277
 - ordering of accesses 305
 - overlapping accesses 288
 - protocol data transmission 274
 - relating with atomic protocol 305
 - sequential consistency 308
 - silent master 263
 - silent slaves 263
 - simultaneously ending accesses 290
 - stable decision 296
- data paths 240
 - data RAM 241
 - state RAM 245
 - tag RAM 243
- hardware configuration 223
- initialization 260
- interfaces 236
- liveness 310
- master arbitration 257
 - fairness 259
- memory bus 238
- MOESI protocol
 - global transactions 231
 - local transactions 231

- shift unit 108, 151
 - implementation 112
- single-core pipelined processor *see*
 - pipelined processor
- single-core sequential processor *see*
 - sequential processor
- SLC *see* cyclic left shifter
- software conditions 133, 176, 191
- spike 46
- SRLC *see* cyclic right-left shifter
- stall engine *see* pipelined processor
- start cycle of an access 279
- SU *see* shift unit
- switching function 26
 - computing by Boolean expression 26
- system of representatives 13
- tree 36
 - o-tree 36
 - OR tree 40
- tristate bus 57
 - bus contention 59, 62
 - control logic 64
 - operation of main memory 72
 - simulation by digital model 61
- tristate driver 56
 - n -bit 68
 - output circuitry 59
- two's complement numbers 18
 - properties 19
- visible registers 167, 168
- zero tester 36