On the realizability of hardware microthreading. Revisiting the general-purpose processor interface: consequences and challenges

Poss, R.C.

Citation for published version (APA):
List of Side Notes

1.1 Fundamental sciences are also application-driven. .......................... 18
1.2 The second rise of separated computing. ........................................ 23
1.3 Embedded/specialized vs. general-purpose computers. ....................... 23
1.4 Example functions that can be made primitive recursive. ................... 24
1.5 Example functions that are not primitive recursive. ........................ 24

3.1 Active, ready and waiting queues. ................................................. 46
3.2 New instruction vs. control bits for thread termination. .................... 49
3.3 Fine-grained thread states. ....................................................... 51
3.4 Using a distributed cache between microthreaded cores. .................... 54
3.5 Sub-events for remote creations. ............................................... 56

4.1 Logical index sequence. ............................................................ 67
4.2 Purpose and motivation of “shared” synchronizers. ............................. 69
4.3 Thread switching not specified by control bits. ................................ 72
4.4 Switching and thread termination as instructions. ............................. 72

6.1 Contenders to C to pitch new architectural designs. .......................... 99

7.1 About implicit communication. ................................................... 116

9.1 Support for heap-based dynamic allocation. ................................... 152

10.1 Static vs. dynamic allocation failure management. ............................ 171

12.1 Implementation-independent interleaving. ....................................... 189

13.1 About the relevance of the Livermore loops. .................................. 205
13.2 Description of the example heterogeneous workload. ......................... 211
13.3 Choice of QuickSort for evaluation. ........................................... 213

E.1 Processor address decoding in the reference implementation. ............... 266
E.2 About the distribution of families with “shared” channels. ................ 266

F.1 About the concept of objects in the C language specification. .............. 270
F.2 About immutable objects. ....................................................... 270
F.3 About objects without an address. ............................................. 270
F.4 About object sizes. .................................................................... 270
F.5 About array item properties. ........................................... 271
F.6 About multiple accesses between sequence points. ..................... 271
F.7 About the initial char representation of objects. .......................... 271
F.8 About valid addresses one past the last char. ............................. 271
F.9 Objects without primary designators. .................................... 273
F.10 Primary designator aliases for immutable objects. ....................... 273
F.11 About array designators in function parameter lists. .................... 274

G.1 Attempt to constrain the well-formedness of programs. ..................... 283
G.2 About the input availability of “shareds” after writes. ................... 286

H.1 About the avoidance of a C syntax parser. ................................ 300
H.2 Preservation of line number markers in M4. ................................. 300

I.1 About the compatibility of our implementation. ............................ 319
I.2 About the syntax of “sl_create” and “sl_sync”. ............................ 328
I.3 About pointers to thread functions. ...................................... 328
I.4 Defining the index sequence in the abstract semantics. .................... 330