Index

$FIB supercombinator, 44
+ combinator, 39

application, 5
argument cell, 15, 105
auto-incrementing pointer access, 27, 31-32

B combinator, 118

C combinator, 119
C programming language, 3, 6, 19, 30-33, 47, 51, 54, 58-59
cache design parameter
- associativity, 64, 75-76, 80, 82
- block size, 63, 68, 73-75, 81-82
- organization, 64
- replacement policy, 63-64, 76
- size, 63-64, 68, 72-73, 75, 79-80
- write allocation, 64, 68, 70-71, 79
- write policy, 63-64, 77, 82

cache memory, 33, 35, 63, 65, 67, 69, 71, 73, 75, 77-79, 82, 84-85, 90-91
cache memory implementation
- split, 35, 64, 69, 77, 79
- unified, 64, 69, 76

cache memory pre-touch, 34
cache simulator, 63-64, 91
case analysis, 10-11, 15, 18, 21, 32, 42
closure reduction, 6-8
closures, **6-9**, 11-14, 55, 59, 98
combinator, 2-3, 7, **118-119**
combinator graph reduction, 2
computational suspension, 7-8, 11, 108

DECstation 3100, 51, 57, 71, 78-79, 82-83, 85, 89-90
delay slot
- branch, 51, 93-94
- load, 30, 35, 51

dummy write, 35

fib benchmark, 44, 50-53, 58, 60, 64, 70, 78, 80, 84, 89-90, 94
first class objects, **6**, 101, 105
Forth programming language, 26-27, 95
function cell, 15, 105
functional programming, 1-2, 5, 101, 121

G-machine, 10, 13, 16-17, 55, 99, 121

garbage collection, **8**, 11-12, 32, 38, 43, 45, 64, 75, 83-86
  generational, 45, 73
  mark/sweep, 45, 85
  stop-and-copy, 45-46, 51, 70, 73, 85
graph reduction, 2, 5-10, 12-15, 19, 23, 26, 28, 36, 45-46, 56, 59, 70-73, 75, 79, 97-98, 101, 121-122

Haskell programming language, 6
heap allocation, 85
Hyperlazy evaluation, 10, 55

I combinator, 37, **103**
IF combinator, 40-41
imperative language, 1, 3, 5-6, 8, 58, 98, 101
indirection node, 17-18, 108

`jsb` opcode, 12, 22-23, 33-34, 46, 55

K combinator, 38, **103**

lambda calculus, 2, 7, 13, 102-103
lazy evaluation, 1, **5-6**, 11, 58, 105
lazy functional programming, 2, 5-9, 58, 98
Lazy ML, 6
LISP programming language, 6, 41, 101
list manipulation combinators, 41
LIT combinator, 39
locality
  spatial, 73, 75, 98
  temporal, 72-73, 75, 80, 82, 91

mapping of TIGRE onto execution models, 31, 33, 35
memoize, 6
MIPS R2000, 25, 30-31, 35-36, 51, 58-59, 75, 82-83, 86, 89-93, 95
Miranda programming language, 6, 9, 54
miss ratio, 64

nfib benchmark, 49-53, 55-56, 58, 60
non-strict combinator, 37-39
NORMA, 12, 56
normal order evaluation, **5**, 8, 11, 105, 109
nRAPS, 53
nthprime benchmark, 50-51, 58-59, 61, 78

P combinator, 42
parallelism, 2, 4, 6, 9, 12, 46, 57-58, 98, 121
Pascal programming language, 6
projection combinators, 37

queens benchmark, 50-51, 78

RAPS, **49**, 52
referential transparency, **5**, 121

S combinator, 103
S’ combinator, **28**, 32-33, 35

SASL programming language, 6
Scheme programming language, 57
self-modifying program techniques, 14, 33, 35-36, 92-93
SK-combinators, 7, 9, **103**
SKI combinator set, 64, 70, **103**, 105, 107, 109, 111, 113, 115
skifib benchmark, 50, 52, 64, 71, 82-83
special hardware support, 7, 9, 12, 14, 23, 56, 59, 89-90, 92, 94, 96-97
spine, **11**, 22-23, 111
spine stack, 16, 19-22, 25-27, 29, 31, 33, 46, 86, 93, **111**
spine stack fixup, 85-86
strict combinator, 38-40, 105, 108
strict evaluation, 6, 57, 98
strictness analysis, 9, 43, 53, 56, 70, 97
subroutine return address stack, 21-23, 25, 33, 86
supercombinator, 3, 9-11, 23, 43, 45, 50-51, 53-56, 59, 70-72, 74-75, 77, 84, 91, 94, 97, 99, 120

tags, 11-12, 15-17, 23, 31, 35, 46, 57
tak benchmark, 50, 52
threaded interpretation, 12, 14, 19, 21, 26, 97
thunk, 6-7
TIGRE, 14-15
TIGRE abstract machine, 5, **25-29**
TIM abstract machine, 11, 13, 56, 98-99, 121
traffic ratio, 64
translation lookaside buffer (TLB), 75, 83
Turner Set, 9, 12, 17, 23, 37-38, 40-41, 43, 51, 59, 64, 70-71, 78, 97, **117**, 119

U combinator, 43

VAX, 3, 16-17, 22-23, 25, 30-31, 33-35, 46, 55, 71, 94
VAX 8800, 33-35, 51, 71, 85
VAXstation 3200, 51, 55
virtual memory, 75, 83