

INDEX

Symbols

3DNow! 428

^ AND 59

¬ NOT 60

∨ OR 59

⊕ XOR 86

* (asterisk) dereference 387

*/ (asterisk, slash) end C comment 27

circuit symbol

capacitor 92

inductor 92

switch 92

. (dot)

select struct field 381, 387

send message in C++ 398

μ (Greek mu) micro 95

Ω (Greek omega) ohms 93

(hash mark) for starting comment 204

-> (hyphen, greater than) dereference
and select field 387

% (percent) C integer division
remainder 41, 365

/ (slash) C integer division quotient
41, 365

/* (slash, asterisk) begin C comment 27

~ (tilde) C++ destructor naming 396

:: (two colons) C++ class
membership 402

// (two slashes) begin C++ comment 397

// (two slashes) floor division 365

A

AC (alternating current) 90

active component 100

adder

for ripple-carry subtraction 119

full 116

full from two halves 117

half 115

ripple-carry 119

adder 115–120

addition

binary 44

decimal 40

addressing modes

direct memory 252

immediate 250

register indirect with indexing
255, 375

register indirect with offset 253

register indirect with scaled
indexing 255, 375

adjacency rule 77

Advanced Vector Extensions (AVX) 428

alternating current (AC) 90

American Standard Code for
Information Interchange
(ASCII) 21

amp 90

ampere 90

arbitrary-precision arithmetic, link to
libraries 440

array

accessing single element 372, 375

character 372

definition of in C 372

element data type 372

in assembly language 376–379

in C 372–376

in C, pass by pointer 373–374

in C, pass by pointer syntax 374

index 372

integer 372

ASCII (American Standard Code for
Information Interchange) 21

- as options
 - a 246
 - gstabs 210
 - l 246
 - o 210
- assembler
 - algorithm 258–260
 - data size notation 239
 - directives 199–203
 - local symbol table 258
 - opcode table 259
 - two-pass 258
- assembler directive (pseudo op) 204
- assembler directives
 - .align 292
 - .ascii 317
 - .asciz 317
 - .bss 291, 311
 - .byte 318
 - .cfi 199
 - .comm 292
 - .equ 230, 411
 - .file 201
 - .globl 202
 - .include 392
 - .int 315, 318
 - .intel_syntax 203
 - .long 315, 318
 - memory allocation 317
 - .quad 285
 - .rodata 226, 230
 - .set 404
 - .size 292
 - .string 317
 - .text 202, 230
 - .type 203, 292
 - .zero 292
- assembly language
 - and machine language 195, 206
 - comments 205
 - compiler-generated 197–201
 - data size directives 207
 - label field 204
 - operand field 204
 - operation field 204
- assembly listing 246
- AT&T syntax 217
- AVX (Advanced Vector Extensions) 428

B

- base 14
- battery 90
- baud 453
- BCD (binary coded decimal) 23
- big endian 35, 36
- binary
 - C syntax 12
 - binary and unsigned decimal 13–17
 - binary coded decimal (BCD) 23
 - binary digit 10
 - binary number system 13
 - binary point 417
 - bitmask 336
 - bit masking 335, 420
 - in assembly language 340–342, 359
 - in C 336–339, 353
- bits 10
 - representing groups of 10–12
- Boolean algebra
 - basic operations of 57
 - literal 60
 - operator precedence 61
 - term 60
- Boolean algebra rules
 - annulment value 63, 65
 - associative 61
 - commutative 63
 - complement 65
 - distributive 63
 - idempotent 65
 - identity value 62
 - involution 63
- Boolean expression minimization 71–86
 - minimal product of sums 72
 - minimal sum of products 72
 - using algebraic manipulations 73
 - using Karnaugh maps 76–84
- Boolean expressions 60–66
- Boolean functions 66–86
 - canonical form 67
 - canonical product 70
 - canonical sum 68
 - conjunctive normal form 70
 - disjunctive normal form 68
 - full conjunctive normal form 70
 - full disjunctive normal form 68

- maxterm 70
- minterm 68
- product of maxterms 70
- product of sums 68
- product term 68
- sum of minterms 68
- sum of products 68
- sum term 69
- Boolean operators 58–60
 - AND 59
 - combining basic 86
 - NOT 60
 - OR 59
 - XOR 86
- Boole, George 57
- bus
 - address 2, 165
 - control 2, 165
 - control in PC 445
 - data 2, 165
 - timing 445
- bus structure, hierarchical 445
- byte 12
 - order in memory 35–36
- C**
- C++
 - calling C functions in 397
 - compiler 398
 - exception 398
 - function overloading 400
 - guidelines, link to 405
 - hidden argument to function 400
 - instantiate an object 396, 398
 - library functions 402
 - object 396–398
 - standard library 25
- C++ class 396
 - and C record 396
 - compiler-generated
 - constructor 405–407
 - compiler-generated
 - destructor 407
 - constructor 396
 - constructor name 396
 - data member 396
 - data member, initialization of 406
 - declaration 397
 - default constructor 396
 - defining member function 401
 - destructor 396
 - destructor name 396
 - inline constructor 407
 - member function 396
 - private scope 397
 - public scope 397
 - struct declaration 397
- C++ Core Guidelines, link to 405
- C++ Crash Course (Lospinoso, Josh) 396
- cache 164, 166–167
 - levels 166–167
 - line 166
 - speed 167
- capacitor 94–97, 175
- carry flag (CF) 39, 44, 186
- central processing unit. *See* CPU
- CET (Control-flow Enforcement Technology) 200
- C function
 - arguments to 28
 - definition 27
 - format 27
 - parameters in 28
 - prototype 27, 288
 - prototype statement 292
- characters 20
- circuit symbols
 - adder 119
 - AND gate 59
 - decoder 122
 - MOSFET 102
 - multiplexer 126
 - NAND gate 107
 - NOR gate 107
 - NOT gate 60
 - OR gate 59
 - PLA 129, 132
 - resistor 92
 - ROM 131
 - tristate buffer 126
 - XOR gate 86
- clock 144
- clock signal 144

- CMOS (complementary metal-oxide semiconductor) 104
 - NAND gate 106
 - power consumption 106
 - switching time 106
- CMOS switch 104–106
- code segment 202
- comparing values 271
 - signed 266
 - unsigned 266
- comparison of canonical Boolean forms 70
- compiler 3
 - assembly language from 198–201
- compiler steps
 - assembly 196
 - compilation 196
 - linking 197
 - preprocessing 196
- complement 45, 60
 - diminished radix 46
 - radix 46
- complementary metal-oxide semiconductor (CMOS) 104
- COM port 452
- computer subsystems 1–2
- conditional branch point 156
- conditional jumps
 - offset 256
 - table of 266
- conductor 90
- conjunction 57, 59
- connected in parallel 94
- connected in series 93
- control flow
 - declarative programming 263
 - fundamental constructs 264
 - imperative programming 263
- Control-flow Enforcement Technology (CET) 200
- control unit 9
- conversion
 - hexadecimal to integer 344, 350
 - integer to signed decimal 370
 - integer to unsigned decimal 361, 368
 - signed decimal to integer 360
 - to uppercase 338, 341
 - unsigned decimal to integer 353, 358
- converting
 - binary to unsigned decimal 15
 - unsigned decimal to binary 16
- coprocessor, floating-point 427
- coulomb 90
- count-controlled loop 273
- C preprocessor directives
 - #define 292, 337
 - #endif 292
 - #ifndef 292
 - #include 292
- C programming 25
 - comments in 27
 - first program 27
- CPU 178–179
 - 32-bit mode 177
 - 64-bit mode 177
 - internal bus 178, 182
 - registers 182
- CPU features, accessing in assembly language 326–332
- C runtime environment 28, 196
- C standard library 25, 27
 - functions
 - gets 336
 - offsetof 391
 - printf 28
 - puts 277, 336
 - scanf 29
 - not using 470–474
 - relationship to operating system 26
 - stdio.h* header file 27
- C-style string 22
- C variable
 - declaration 288
 - definition 288
 - global 290
 - local 288
 - name scope 288
 - scope and lifetime 314

D

- data segment 202
- data size
 - byte 183, 187
 - doubleword 183, 187
 - instruction for extending 270
 - quadword 183, 187
 - word 183
- data types, integral 187–188
- DC (direct current) 90
- debugger, examining memory 30
- decimal number system 10
- decimal point 417
- decimal system 10
- decoder 173, 121–123
- decoder ring 53
- decToSInt 360
- decToUInt 358
- De Morgan's law 65
- Design and Evolution of C++, The*
(Stroustrup, Bjarne) 396
- device controller 446
- D flip-flop 168, 170–171
- direct current (DC) 90
- direct memory access (DMA) 450
- disjunction 57, 59
- division, integer 360–369
 - avoiding 369
 - by powers of two 351
 - quotient 365, 369
 - register usage 365
 - remainder 365, 369
 - signed versus unsigned 365
- DMA (direct memory access) 450
 - controller 450
- do-while loop 274
 - compared to while and for 275
- DRAM (dynamic random-access memory) 175
- duality 66
- dynamic random-access memory (DRAM) 175

E

- editors 2, 5
- effective address 226, 253
- electric field 94
- electronics 90–100
- ELF (Executable and Linking Format) 202
- endianness 35
- energy 99
- erasable programmable read-only memory (EPROM) 131
- exception 466, 469
 - CPU response to 467–469
 - handler 467
 - page fault 448
 - terminology 468
- executable file 197
- Executable and Linking Format (ELF) 202
- external interrupt 466, 468

F

- farads 95
- fast system call 472
- fetching 3
- file descriptors
 - STDERR_FILENO 222
 - STDIN_FILENO 222
 - STDOUT_FILENO 222
- filename extensions 197
- finite state machine 136
- fixed-point numbers 417–424
 - fraction in decimal 421
 - fraction in powers of two 417
- flip-flop, 144–152
 - asynchronous input 146
 - D 145
 - JK 148, 155, 160
 - primary portion 145, 149
 - negative-edge triggering 146
 - positive-edge triggering 146
 - secondary portion 145, 149
 - T 147

- floating-point arithmetic error 433–440
 - absorption 435
 - associativity 437
 - cancellation 436
 - man fenv 430
 - rounding 433
 - floating-point numbers 425–427
 - not real numbers 425
 - programming with 430
 - rounding mode 430
 - floating-point representation 425
 - biased exponent 427
 - double 426
 - exponent 425
 - float 426
 - hidden bit 427
 - significand 425
 - x86-64 extended version 426
 - floor division, in Python 365
 - for loop 272
 - fractional values in binary 416
 - frame pointer 185, 209, 216, 298, 307
 - front-side bus 445
 - function
 - epilogue 210
 - epilogue, inside 232
 - input to 289
 - minimal processing 208–210
 - output from 289
 - prologue 210
 - prologue, inside 231
 - return value 294, 338
 - function arguments 289
 - more than six 299–306
 - pass by pointer 294
 - pass by reference 294
 - pass by value 294
 - passing in C 294–303
 - pass in registers 223–224
 - pushing onto the stack 299
 - in registers 296–298
 - return value 294
 - storing directly on the stack 303
- G**
- gate
 - AND 59
 - NOT 60
 - OR 59
 - XOR 86
 - gate descriptor 466
 - gcc options
 - c 196
 - E 196
 - fcf-protection=none 201
 - fno-asynchronous-unwind
 - tables 199
 - masm=intel 198
 - o 197
 - O0 198
 - S 196, 198
 - gdb 30
 - as a learning tool 31
 - breakpoint 32
 - commands
 - b 31
 - c 31
 - h 31
 - i r 31
 - l 31
 - layout regs 212
 - n 189
 - q 35
 - r 31
 - s 189
 - set disassembly-flavor 211
 - si 189
 - tui enable 212
 - x 31
 - gdb debugger
 - learning assembly
 - language 210–216
 - TUI mode 211–217
 - viewing registers 188–193
 - viewing stack frame 231–233
 - getInt 370
 - global offset table (GOT) 227, 260
 - global variables 290–293
 - Goldberg, David (“What Every Computer Scientist Should Know About Floating-Point Arithmetic”) 440
 - GOT (global offset table) 227, 260
 - Gray code and Karnugh maps 79
 - Gray, Frank 79

H

- handler 467
- Harvard architecture 166
 - in cache 167
- header file 27, 292
- heap segment 202
- henrys 97
- hexadecimal 10
 - C syntax 12
- hexadecimal characters, UTF-8 code
 - for 21
- hexadecimal digit 10
 - four bits 10, 43
 - signed decimal 47
 - two's complement 47
 - unsigned decimal 43
 - using 12

I

- IDT (interrupt descriptor table) 467
- IEEE 754 floating-point standard 426
 - biased exponent 427
 - hidden bit 427
 - link to 426
 - normalized form 426
- if conditional 276–278
- if-else ladder 282
- if-then-else conditional 278
- inb function 463
- inductor 97–99
- information hiding 294
- inline assembly language 332–334
- instruction
 - execution cycle 180–181
 - fetch 181
 - general format 206
 - operation code (opcode) 204
 - pointer 180
 - queue 180
 - register 180
- instruction bytes. *See* machine code
- instructions
 - add 236
 - addss 432
 - and 339
 - call 226
 - cbw 375
 - cdqe 375
 - cmp 271
 - cvtss2sd 432
 - cwde 375
 - div 364
 - endbr64 200
 - idiv 364
 - imul 355
 - in 447
 - inc 272
 - int 470
 - iret 470
 - jcc 265
 - jcxz 265
 - je 238
 - jecxz 265
 - jmp 264
 - jrcxz 265
 - lea 226
 - leave 239
 - mov 206
 - movss 432
 - movsx 347
 - movsxd 347
 - movzx 270
 - mul 356
 - neg 237
 - nop 298
 - or 340
 - out 447
 - pop 208
 - push 208
 - ret 208
 - sal 347
 - sar 348
 - setcc 330
 - shl 349
 - shr 348
 - sign-extend for signed division 365
 - sub 236
 - syscall 472
 - test 270
 - xor 238
- int 0x80
 - Linux operations 471
 - register usage 471
 - software interrupt 470
- integer codes, circular nature of 53
- integer unit 428

- integral data types 183
- integral values 415
- interpreter 3
- interrupt controller 449
- interrupt descriptor table (IDT) 467
- interrupt-driven I/O 449
- interrupt handler 450, 467
- interrupt
 - CPU response to 467–469
 - handler 450, 467
 - terminology 468
- intToSDec 370
- intToUDec 368
- invert 60
- I/O controller hub 445
- I/O controller register
 - control 446
 - receive 446
 - status 446
 - transmit 446
- I/O devices 443
 - accessing 446–449
- I/O functions
 - decToSInt 360
 - decToUInt 358
 - hexToInt 360
 - getInt 370
 - intToSDec 370
 - intToUDec 368
 - putInt 370
 - readLn 308
 - writeStr 308
- io.h* header file 461
- I/O, memory-mapped 447
- I/O, port-mapped 447
- I/O ports 447
- I/O programming 449
- isolated I/O 447
- iteration 264, 267–275
 - versus recursion 267, 320

J

- joule 99
- jump
 - conditional 265–267
 - long 256

- near 256
- short 256
- unconditional 264
- jump instructions 256–257
- jump table 283

K

- Kahan summation algorithm, link to 440
- Karnaugh, Maurice 76
- Karnaugh map 76–86, 154, 160

L

- latch 136–144
 - D 142
 - feedback in 136
 - SR, gated 141
 - SR using NAND gates 139
 - SR using NOR gates 136
 - SR with Enable 141
- ld options
 - e 472
 - o 472
- leaf functions 298
- least significant digit 14
- linker
 - algorithm 260
 - global symbol table 260
- listing file, assembler 246
- little-endian 34
- locality of reference 167
- logic circuit 113
 - combinational 114
 - sequential 114
- loop control variable 267
- looping 267–275
- Lospinoso, Josh (*C++ Crash Course*)
 - 396, 406

M

- machine code
 - looking at 246
 - ModR/M byte 248
 - opcode bytes 247
 - operands 247
 - REX prefix byte 250
 - SIB byte 255

- magnetic field 97
- main function 28, 221
- main memory 165–166, 175
 - organization 166
- mask 336
- mass storage 164–165
- Mealy state machine 136
- memory
 - addresses 19
 - cache 164
 - cost 164
 - data storage 18–23
 - hardware 168–176
 - hierarchy 163–164
 - layers 164
 - main 164–166, 175
 - nonvolatile 164
 - offline 164
 - page 448
 - page frame 448
 - page map table 448
 - physical 448
 - random access
 - dynamic 175–176
 - static 173–175
 - read-write 172–173
 - speed 164
 - timing 444
 - virtual 448
 - volatile 165
- memory controller hub 445
- memory-mapped I/O 447
 - in assembly language 457–459
 - in C 452–457
- memory mapping unit 448
- memory segments
 - bss 292
 - characteristics 317
 - data 202
 - heap 202
 - stack 202
 - text 202
- metal-oxide-semiconductor field-effect transistor (MOSFET) 101
- minimum function
 - assembly language 208
 - C 197

- Moore state machine 136
- MOSFET (metal-oxide-semiconductor field-effect transistor) 101
 - channel 101
 - drain 101
 - gate 101
 - N-channel 102
 - P-channel 103
 - power consumption 104
 - source 101
 - switch 101–104
 - switching time 104
- most significant digit 14
- multiplexer (MUX) 124–127, 172
- multiplication, integer 352–359
 - by powers of two 351
 - register usage 356
 - signed versus unsigned 356
- MUX (multiplexer) 124–127, 172
- mxcsr register 429

N

- name mangling 313, 400
- NAND gate, universal 108–110
- negation 48, 57
- newline character 23
- NOR gate, universal 110
- northbridge 445
- NOT gate 105
- numerical accuracy 440

O

- object
 - attribute 395
 - instance 395
 - instantiate 396
 - message 395
 - method 395
 - using in C++ 398–400
- object file 196, 202, 210
- objects in assembly language 407–412
- octal 11
 - C syntax 12
- octal digit, three bits 12
- ohms 92
- Ohm's law 92
- one's complement 49

- organizing data 371
- outb function 463
- overflow flag (OF) 39, 50–53, 186

P

- page fault exception 448, 469
- PAL (programmable array logic) 131
- Pascal 23
- passive component 92
- permissions, file 466
- pipeline 156
- PLA (programmable logic array) 128
- PLD (programmable logic device) 127–132
- PLT (procedure linkage table) 226
- polled I/O 449
 - programming algorithms 450
- port-mapped I/O 447, 460–461
- positional notation 13
- position-independent code 225
- position-independent executable 225
- potential difference 90
- power 99–100
- power supply 90
- principle of duality 66
- printf, conversion specifiers 29
- privilege levels, CPU 466
- procedure linkage table (PLT) 226, 260
- program execution 2–3
- programmable array logic (PAL) 131
- programmable logic array (PLA) 128
- programmable logic device (PLD) 127–132
- programmable read-only memory (PROM) 131
- programmed I/O 449
- programming documentation
 - info 4
 - man page 4
- programming environment 4–6
- PROM (programmable read-only memory) 131
- propagation delay 143
- pseudo op 204
- pull-down device 103
- pull-up device 103
- putInt 370

R

- radix 14
- radix point 417
- RAM (random access memory) 18, 165
- rbp register 185
- read-only memory (ROM) 19, 130
- real numbers 425
- record 380
 - in assembly language 382
 - in C 380–382
 - element 380
 - field 380
 - layout in memory 382
 - member 380
 - pass by pointer advantage 389–393
 - pass by value in C 386
 - pass in assembly language 389–393
 - passing in C 383–389
- recursion 319–326
 - base case 320
 - compared to iteration 267, 320
 - save registers 322
 - stack usage 323–326
- register 168, 182–186
 - 32-bit 184, 207
 - 64-bit mode 184
 - bit numbering 183
 - file 172
 - general purpose 183–186
 - general purpose, naming 183, 185
 - general purpose, usage 224
 - hardware implementation 168–170
 - mxcsr 429
 - passing arguments in 224
 - rbp 307
 - rflags 39, 186
 - rsp 307
 - shift 171–172
 - sizes 183
 - status 186
 - xmm 428
 - ymm 428
 - zmm 428
- register content, saving 224
- resistor 92–94

- rflags register, 186, 467–468
 - status flags 39, 265
- rip register 180
- rip-relative addressing 180
- ROM (read-only memory) 19, 130
- Roman numerals 14
- rsp register 185

S

- SAM (sequential access memory) 19
- scanf 29
 - conversion specifiers 29
- scientific notation 425
- selection 264, 276–285
- semiconductor 101
 - doping 101
 - holes 101
 - N-type 101
 - P-type 101
- sentinel character 22
- sentinel value 22
 - loop control 272
- sequential access memory (SAM) 19
- sequential logic circuit, designing 151
 - branch predictor 156
 - counter 152
- settling time 143
- Shannon, Claude 58
- shifting bits 343
 - in assembly language 349–351
 - in C 343–347
- shift instructions 347
- signal voltage levels, active-high and active-low 114
- sign bit 47
- signed integers 15
 - addition 50–53
 - subtraction 50–53
- sign flag (SF) 186
- sign-magnitude code 45
- SIMD (single instruction, multiple data) 428
- SoC (system on a chip) 446
- software interrupt 466, 469
- source code 2
- southbridge 445

- SRAM 173
- SSE (Streaming SIMD Extension) 428
- SSE2 floating-point hardware 427–430
 - rounding mode 430
 - status and control register 429
- stack
 - ascending 228
 - corruption 237
 - data structure 227–229
 - descending 228
 - empty 229
 - full 228
 - red zone 298, 303, 342
 - segment 202
- stack canary 237–238
- stack frame 209, 234–237, 298
 - usage 306–307
- stack pointer 185, 207, 216, 307
 - address boundary 298, 301
 - addressing boundary 281
 - alignment 236
 - local variables 236
 - moving 303
- standard error 222
- standard in 222
- standard out 222
- state diagram 136, 140, 152, 157
- state, system 135
- state transition table 138, 140, 142, 152, 159
- static random-access memory (SRAM) 173
- status flags
 - carry flag (CF) 39, 44, 119, 186
 - overflow flag (OF) 39, 50, 119, 186
 - sign flag (SF) 186
 - zero flag (ZF) 186
- status register 186
- STDIN_FILENO 222
- STDOUT_FILENO 222
- STDERR_FILENO 222
- storage blocks
 - data 165
 - instructions 165
- Streaming SIMD Extension (SSE) 428
- Stroustrup, Bjarne (*Design and Evolution of C++, The*) 396, 405

- struct
 - accessing field in C 381
 - declaring new data type of 384
 - defining in C 381
 - defining with tag in C 385
 - field memory alignment 391
 - pass by value in C 386, 388
 - tag 383–386
 - tag for passing as argument 386
- structure 380
- subfunctions 221
- subtraction
 - binary 44
 - borrow in 41
 - decimal 41
- subtractor 120
- supervisor mode 466
- Sutter, Herb 405
- switch 92
- switch conditional 282–285
- switches, representing 10–12
- switching algebra 58
- syscall (system call) 470, 472
 - Linux operations 473
 - register usage 473
- system call. *See* syscall
- system call functions
 - read 222
 - write 222, 267
- system on a chip (SoC) 446

T

- testing bits 270
- text segment 202
- text string 22
- this pointer 404, 409, 411
- time constant 96, 98
- timing considerations 444
 - bus 445
 - I/O device 444
 - memory 444
- toggle 147
- transistor 100–106
 - switching time 106

- tristate buffer 125, 172
- truth table 58
 - AND 59
 - NOT 60
 - OR 59
 - XOR 86
- two's complement 45–49
 - computing 48
- type casting 347, 350, 355, 359, 363

U

- UART (universal asynchronous receiver/transmitter) 450
 - 16550 data sheet, link to 453
 - 16550 registers 451
 - data bits 450
 - port-mapped 460
 - programming in assembly
 - language 457
 - programming in C 452
 - start bit 450
 - stop bit 450
- Unicode UTF-8 20
- universal asynchronous receiver/transmitter (UART) 450
- unsigned integers 15, 23
 - addition 40–45
 - subtraction 40–45
- uppercase versus lowercase, ASCII 336
- user mode 466
- UTF-8 20

V

- variable
 - automatic 233
 - local 233–237, 316
 - static local 309–316
- vector 467
- vector table 467
- very-large-scale integration (VLSI) 146
- volt 90
- von Neumann
 - architecture 166
 - bottleneck 166
- von Neumann, John 166

W

watt 99
“What Every Computer Scientist
Should Know About Floating-
Point Arithmetic” (Goldberg,
David) 440
while loop 267–272

X

x87 floating point unit 427

Z

zero flag (ZF) 186