

# VB41CX Quick Reference Guide

The image displays the VB41CX Quick Reference Guide, which is a screenshot of a software application window titled "VB41CX REFERENCE". The window shows a list of functions organized into columns. To the right of the list is a photograph of the physical HP41C calculator, which is a handheld device with a numeric keypad and a small display screen. Below the calculator image is a small inset showing a list of functions, likely a continuation of the main list.

**Function List:**

- 01+LBL 00
- 02+LBL 01
- 03+LBL 02
- 04+LBL 03
- 05+LBL 04
- 06+LBL 05
- 07+LBL 06
- 08+LBL 07
- 09+LBL 08
- 10+LBL 09
- 11+LBL 10
- 12+LBL 11
- 13+LBL 12
- 14+LBL 13
- 15+LBL 14
- 16+LBL 15
- 17+LBL 16
- 18+LBL 17
- 19+LBL 18
- 20+LBL 19
- 21+LBL 20
- 22+LBL 21
- 23+LBL 22
- 24+LBL 23
- 25+LBL 24
- 26+LBL 25
- 27+LBL 26
- 28+LBL 27
- 29+LBL 28
- 30+LBL 29
- 31+LBL 30
- 32+LBL 31
- 33+LBL 32
- 34+LBL 33
- 35+LBL 34
- 36+LBL 35
- 37+LBL 36
- 38+LBL 37
- 39+LBL 38
- 40+LBL 39
- 41+LBL 40
- 42+LBL 41
- 43+LBL 42
- 44+LBL 43
- 45+LBL 44
- 46+LBL 45
- 47+LBL 46
- 48+LBL 47
- 49+LBL 48
- 50+LBL 49
- 51+LBL 50
- 52+LBL 51
- 53+LBL 52
- 54+LBL 53
- 55+LBL 54
- 56+LBL 55
- 57+LBL 56
- 58+LBL 57
- 59+LBL 58
- 60+LBL 59
- 61+LBL 60
- 62+LBL 61
- 63+LBL 62
- 64+LBL 63
- 65+LBL 64
- 66+LBL 65
- 67+LBL 66
- 68+LBL 67
- 69+LBL 68
- 70+LBL 69
- 71+LBL 70
- 72+LBL 71
- 73+LBL 72
- 74+LBL 73
- 75+LBL 74
- 76+LBL 75
- 77+LBL 76
- 78+LBL 77
- 79+LBL 78
- 80+LBL 79
- 81+LBL 80
- 82+LBL 81
- 83+LBL 82
- 84+LBL 83
- 85+LBL 84
- 86+LBL 85
- 87+LBL 86
- 88+LBL 87
- 89+LBL 88
- 90+LBL 89
- 91+LBL 90
- 92+LBL 91
- 93+LBL 92
- 94+LBL 93
- 95+LBL 94
- 96+LBL 95
- 97+LBL 96
- 98+LBL 97
- 99+LBL 98
- 100+LBL 99
- 101+LBL 100
- 102+LBL 101
- 103+LBL 102
- 104+LBL 103
- 105+LBL 104
- 106+LBL 105
- 107+LBL 106
- 108+LBL 107
- 109+LBL 108
- 110+LBL 109
- 111+LBL 110
- 112+LBL 111
- 113+LBL 112
- 114+LBL 113
- 115+LBL 114
- 116+LBL 115
- 117+LBL 116
- 118+LBL 117
- 119+LBL 118
- 120+LBL 119
- 121+LBL 120
- 122+LBL 121
- 123+LBL 122
- 124+LBL 123
- 125+LBL 124
- 126+LBL 125
- 127+LBL 126
- 128+LBL 127
- 129+LBL 128
- 130+LBL 129
- 131+LBL 130
- 132+LBL 131
- 133+LBL 132
- 134+LBL 133
- 135+LBL 134
- 136+LBL 135
- 137+LBL 136
- 138+LBL 137
- 139+LBL 138
- 140+LBL 139
- 141+LBL 140
- 142+LBL 141
- 143+LBL 142
- 144+LBL 143
- 145+LBL 144
- 146+LBL 145
- 147+LBL 146
- 148+LBL 147
- 149+LBL 148
- 150+LBL 149
- 151+LBL 150
- 152+LBL 151
- 153+LBL 152
- 154+LBL 153
- 155+LBL 154
- 156+LBL 155
- 157+LBL 156
- 158+LBL 157
- 159+LBL 158
- 160+LBL 159
- 161+LBL 160
- 162+LBL 161
- 163+LBL 162
- 164+LBL 163
- 165+LBL 164
- 166+LBL 165
- 167+LBL 166
- 168+LBL 167
- 169+LBL 168
- 170+LBL 169
- 171+LBL 170
- 172+LBL 171
- 173+LBL 172
- 174+LBL 173
- 175+LBL 174
- 176+LBL 175
- 177+LBL 176
- 178+LBL 177
- 179+LBL 178
- 180+LBL 179
- 181+LBL 180
- 182+LBL 181
- 183+LBL 182
- 184+LBL 183
- 185+LBL 184
- 186+LBL 185
- 187+LBL 186
- 188+LBL 187
- 189+LBL 188
- 190+LBL 189
- 191+LBL 190
- 192+LBL 191
- 193+LBL 192
- 194+LBL 193
- 195+LBL 194
- 196+LBL 195
- 197+LBL 196
- 198+LBL 197
- 199+LBL 198
- 200+LBL 199
- 201+LBL 200
- 202+LBL 201
- 203+LBL 202
- 204+LBL 203
- 205+LBL 204
- 206+LBL 205
- 207+LBL 206
- 208+LBL 207
- 209+LBL 208
- 210+LBL 209
- 211+LBL 210
- 212+LBL 211
- 213+LBL 212
- 214+LBL 213
- 215+LBL 214
- 216+LBL 215
- 217+LBL 216
- 218+LBL 217
- 219+LBL 218
- 220+LBL 219
- 221+LBL 220
- 222+LBL 221
- 223+LBL 222
- 224+LBL 223
- 225+LBL 224
- 226+LBL 225
- 227+LBL 226
- 228+LBL 227
- 229+LBL 228
- 230+LBL 229
- 231+LBL 230
- 232+LBL 231
- 233+LBL 232
- 234+LBL 233
- 235+LBL 234
- 236+LBL 235
- 237+LBL 236
- 238+LBL 237
- 239+LBL 238
- 240+LBL 239
- 241+LBL 240
- 242+LBL 241
- 243+LBL 242
- 244+LBL 243
- 245+LBL 244
- 246+LBL 245
- 247+LBL 246
- 248+LBL 247
- 249+LBL 248
- 250+LBL 249
- 251+LBL 250
- 252+LBL 251
- 253+LBL 252
- 254+LBL 253
- 255+LBL 254
- 256+LBL 255
- 257+LBL 256
- 258+LBL 257
- 259+LBL 258
- 260+LBL 259
- 261+LBL 260
- 262+LBL 261
- 263+LBL 262
- 264+LBL 263
- 265+LBL 264
- 266+LBL 265
- 267+LBL 266
- 268+LBL 267
- 269+LBL 268
- 270+LBL 269
- 271+LBL 270
- 272+LBL 271
- 273+LBL 272
- 274+LBL 273
- 275+LBL 274
- 276+LBL 275
- 277+LBL 276
- 278+LBL 277
- 279+LBL 278
- 280+LBL 279
- 281+LBL 280
- 282+LBL 281
- 283+LBL 282
- 284+LBL 283
- 285+LBL 284
- 286+LBL 285
- 287+LBL 286
- 288+LBL 287
- 289+LBL 288
- 290+LBL 289
- 291+LBL 290
- 292+LBL 291
- 293+LBL 292
- 294+LBL 293
- 295+LBL 294
- 296+LBL 295
- 297+LBL 296
- 298+LBL 297
- 299+LBL 298
- 300+LBL 299
- 301+LBL 300
- 302+LBL 301
- 303+LBL 302
- 304+LBL 303
- 305+LBL 304
- 306+LBL 305
- 307+LBL 306
- 308+LBL 307
- 309+LBL 308
- 310+LBL 309
- 311+LBL 310
- 312+LBL 311
- 313+LBL 312
- 314+LBL 313
- 315+LBL 314
- 316+LBL 315
- 317+LBL 316
- 318+LBL 317
- 319+LBL 318
- 320+LBL 319
- 321+LBL 320
- 322+LBL 321
- 323+LBL 322
- 324+LBL 323
- 325+LBL 324
- 326+LBL 325
- 327+LBL 326
- 328+LBL 327
- 329+LBL 328
- 330+LBL 329
- 331+LBL 330
- 332+LBL 331
- 333+LBL 332
- 334+LBL 333
- 335+LBL 334
- 336+LBL 335
- 337+LBL 336
- 338+LBL 337
- 339+LBL 338
- 340+LBL 339
- 341+LBL 340
- 342+LBL 341
- 343+LBL 342
- 344+LBL 343
- 345+LBL 344
- 346+LBL 345
- 347+LBL 346
- 348+LBL 347
- 349+LBL 348
- 350+LBL 349
- 351+LBL 350
- 352+LBL 351
- 353+LBL 352
- 354+LBL 353
- 355+LBL 354
- 356+LBL 355
- 357+LBL 356
- 358+LBL 357
- 359+LBL 358
- 360+LBL 359
- 361+LBL 360
- 362+LBL 361
- 363+LBL 362
- 364+LBL 363
- 365+LBL 364
- 366+LBL 365
- 367+LBL 366
- 368+LBL 367
- 369+LBL 368
- 370+LBL 369
- 371+LBL 370
- 372+LBL 371
- 373+LBL 372
- 374+LBL 373
- 375+LBL 374
- 376+LBL 375
- 377+LBL 376
- 378+LBL 377
- 379+LBL 378
- 380+LBL 379
- 381+LBL 380
- 382+LBL 381
- 383+LBL 382
- 384+LBL 383
- 385+LBL 384
- 386+LBL 385
- 387+LBL 386
- 388+LBL 387
- 389+LBL 388
- 390+LBL 389
- 391+LBL 390
- 392+LBL 391
- 393+LBL 392
- 394+LBL 393
- 395+LBL 394
- 396+LBL 395
- 397+LBL 396
- 398+LBL 397
- 399+LBL 398
- 400+LBL 399
- 401+LBL 400
- 402+LBL 401
- 403+LBL 402
- 404+LBL 403
- 405+LBL 404
- 406+LBL 405
- 407+LBL 406
- 408+LBL 407
- 409+LBL 408
- 410+LBL 409
- 411+LBL 410
- 412+LBL 411
- 413+LBL 412
- 414+LBL 413
- 415+LBL 414
- 416+LBL 415
- 417+LBL 416
- 418+LBL 417
- 419+LBL 418
- 420+LBL 419
- 421+LBL 420
- 422+LBL 421
- 423+LBL 422
- 424+LBL 423
- 425+LBL 424
- 426+LBL 425
- 427+LBL 426
- 428+LBL 427
- 429+LBL 428
- 430+LBL 429
- 431+LBL 430
- 432+LBL 431
- 433+LBL 432
- 434+LBL 433
- 435+LBL 434
- 436+LBL 435
- 437+LBL 436
- 438+LBL 437
- 439+LBL 438
- 440+LBL 439
- 441+LBL 440
- 442+LBL 441
- 443+LBL 442
- 444+LBL 443
- 445+LBL 444
- 446+LBL 445
- 447+LBL 446
- 448+LBL 447
- 449+LBL 448
- 450+LBL 449
- 451+LBL 450
- 452+LBL 451
- 453+LBL 452
- 454+LBL 453
- 455+LBL 454
- 456+LBL 455
- 457+LBL 456
- 458+LBL 457
- 459+LBL 458
- 460+LBL 459
- 461+LBL 460
- 462+LBL 461
- 463+LBL 462
- 464+LBL 463
- 465+LBL 464
- 466+LBL 465
- 467+LBL 466
- 468+LBL 467
- 469+LBL 468
- 470+LBL 469
- 471+LBL 470
- 472+LBL 471
- 473+LBL 472
- 474+LBL 473
- 475+LBL 474
- 476+LBL 475
- 477+LBL 476
- 478+LBL 477
- 479+LBL 478
- 480+LBL 479
- 481+LBL 480
- 482+LBL 481
- 483+LBL 482
- 484+LBL 483
- 485+LBL 484
- 486+LBL 485
- 487+LBL 486
- 488+LBL 487
- 489+LBL 488
- 490+LBL 489
- 491+LBL 490
- 492+LBL 491
- 493+LBL 492
- 494+LBL 493
- 495+LBL 494
- 496+LBL 495
- 497+LBL 496
- 498+LBL 497
- 499+LBL 498
- 500+LBL 499
- 501+LBL 500
- 502+LBL 501
- 503+LBL 502
- 504+LBL 503
- 505+LBL 504
- 506+LBL 505
- 507+LBL 506
- 508+LBL 507
- 509+LBL 508
- 510+LBL 509
- 511+LBL 510
- 512+LBL 511
- 513+LBL 512
- 514+LBL 513
- 515+LBL 514
- 516+LBL 515
- 517+LBL 516
- 518+LBL 517
- 519+LBL 518
- 520+LBL 519
- 521+LBL 520
- 522+LBL 521
- 523+LBL 522
- 524+LBL 523
- 525+LBL 524
- 526+LBL 525
- 527+LBL 526
- 528+LBL 527
- 529+LBL 528
- 530+LBL 529
- 531+LBL 530
- 532+LBL 531
- 533+LBL 532
- 534+LBL 533
- 535+LBL 534
- 536+LBL 535
- 537+LBL 536
- 538+LBL 537
- 539+LBL 538
- 540+LBL 539
- 541+LBL 540
- 542+LBL 541
- 543+LBL 542
- 544+LBL 543
- 545+LBL 544
- 546+LBL 545
- 547+LBL 546
- 548+LBL 547
- 549+LBL 548
- 550+LBL 549
- 551+LBL 550
- 552+LBL 551
- 553+LBL 552
- 554+LBL 553
- 555+LBL 554
- 556+LBL 555
- 557+LBL 556
- 558+LBL 557
- 559+LBL 558
- 560+LBL 559
- 561+LBL 560
- 562+LBL 561
- 563+LBL 562
- 564+LBL 563
- 565+LBL 564
- 566+LBL 565
- 567+LBL 566
- 568+LBL 567
- 569+LBL 568
- 570+LBL 569
- 571+LBL 570
- 572+LBL 571
- 573+LBL 572
- 574+LBL 573
- 575+LBL 574
- 576+LBL 575
- 577+LBL 576
- 578+LBL 577
- 579+LBL 578
- 580+LBL 579
- 581+LBL 580
- 582+LBL 581
- 583+LBL 582
- 584+LBL 583
- 585+LBL 584
- 586+LBL 585
- 587+LBL 586
- 588+LBL 587
- 589+LBL 588
- 590+LBL 589
- 591+LBL 590
- 592+LBL 591
- 593+LBL 592
- 594+LBL 593
- 595+LBL 594
- 596+LBL 595
- 597+LBL 596
- 598+LBL 597
- 599+LBL 598
- 600+LBL 599
- 601+LBL 600
- 602+LBL 601
- 603+LBL 602
- 604+LBL 603
- 605+LBL 604
- 606+LBL 605
- 607+LBL 606
- 608+LBL 607
- 609+LBL 608
- 610+LBL 609
- 611+LBL 610
- 612+LBL 611
- 613+LBL 612
- 614+LBL 613
- 615+LBL 614
- 616+LBL 615
- 617+LBL 616
- 618+LBL 617
- 619+LBL 618
- 620+LBL 619
- 621+LBL 620
- 622+LBL 621
- 623+LBL 622
- 624+LBL 623
- 625+LBL 624
- 626+LBL 625
- 627+LBL 626
- 628+LBL 627
- 629+LBL 628
- 630+LBL 629
- 631+LBL 630
- 632+LBL 631
- 633+LBL 632
- 634+LBL 633
- 635+LBL 634
- 636+LBL 635
- 637+LBL 636
- 638+LBL 637
- 639+LBL 638
- 640+LBL 639
- 641+LBL 640
- 642+LBL 641
- 643+LBL 642
- 644+LBL 643
- 645+LBL 644
- 646+LBL 645
- 647+LBL 646
- 648+LBL 647
- 649+LBL 648
- 650+LBL 649
- 651+LBL 650
- 652+LBL 651
- 653+LBL 652
- 654+LBL 653
- 655+LBL 654
- 656+LBL 655
- 657+LBL 656
- 658+LBL 657
- 659+LBL 658
- 660+LBL 659
- 661+LBL 660
- 662+LBL 661
- 663+LBL 662
- 664+LBL 663
- 665+LBL 664
- 666+LBL 665
- 667+LBL 666
- 668+LBL 667
- 669+LBL 668
- 670+LBL 669
- 671+LBL 670
- 672+LBL 671
- 673+LBL 672
- 674+LBL 673
- 675+LBL 674
- 676+LBL 675
- 677+LBL 676
- 678+LBL 677
- 679+LBL 678
- 680+LBL 679
- 681+LBL 680
- 682+LBL 681
- 683+LBL 682
- 684+LBL 683
- 685+LBL 684
- 686+LBL 685
- 687+LBL 686
- 688+LBL 687
- 689+LBL 688
- 690+LBL 689
- 691+LBL 690
- 692+LBL 691
- 693+LBL 692
- 694+LBL 693
- 695+LBL 694
- 696+LBL 695
- 697+LBL 696
- 698+LBL 697
- 699+LBL 698
- 700+LBL 699
- 701+LBL 700
- 702+LBL 701
- 703+LBL 702
- 704+LBL 703
- 705+LBL 704
- 706+LBL 705
- 707+LBL 706
- 708+LBL 707
- 709+LBL 708
- 710+LBL 709
- 711+LBL 710
- 712+LBL 711
- 713+LBL 712
- 714+LBL 713
- 715+LBL 714
- 716+LBL 715
- 717+LBL 716
- 718+LBL 717
- 719+LBL 718
- 720+LBL 719
- 721+LBL 720
- 722+LBL 721
- 723+LBL 722
- 724+LBL 723
- 725+LBL 724
- 726+LBL 725
- 727+LBL 726
- 728+LBL 727
- 729+LBL 728
- 730+LBL 729
- 731+LBL 730
- 732+LBL 731
- 733+LBL 732
- 734+LBL 733
- 735+LBL 734
- 736+LBL 735
- 737+LBL 736
- 738+LBL 737
- 739+LBL 738
- 740+LBL 739
- 741+LBL 740
- 742+LBL 741
- 743+LBL 742
- 744+LBL 743
- 745+LBL 744
- 746+LBL 745
- 747+LBL 746
- 748+LBL 747
- 749+LBL 748
- 750+LBL 749
- 751+LBL 750
- 752+LBL 751
- 753+LBL 752
- 754+LBL 753
- 755+LBL 754
- 756+LBL 755
- 757+LBL 756
- 758+LBL 757
- 759+LBL 758
- 760+LBL 759
- 761+LBL 760
- 762+LBL 761
- 763+LBL 762
- 764+LBL 763
- 765+LBL 764
- 766+LBL 765
- 767+LBL 766
- 768+LBL 767
- 769+LBL 768
- 770+LBL 769
- 771+LBL 770
- 772+LBL 771
- 773+LBL 772
- 774+LBL 773
- 775+LBL 774
- 776+LBL 775
- 777+LBL 776
- 778+LBL 777
- 779+LBL 778
- 780+LBL 779
- 781+LBL 780
- 782+LBL 781
- 783+LBL 782
- 784+LBL 783
- 785+LBL 784
- 786+LBL 785
- 787+LBL 786
- 788+LBL 787
- 789+LBL 788
- 790+LBL 789
- 791+LBL 790
- 792+LBL 791
- 793+LBL 792
- 794+LBL 793
- 795+LBL 794
- 796+LBL 795
- 797+LBL 796
- 798+LBL 797
- 799+LBL 798
- 800+LBL 799
- 801+LBL 800
- 802+LBL 801
- 803+LBL 802
- 804+LBL 803
- 805+LBL 804
- 806+LBL 805
- 807+LBL 806
- 808+LBL 807
- 809+LBL 808
- 810+LBL 809
- 811+LBL 810
- 812+LBL 811
- 813+LBL 812
- 814+LBL 813
- 815+LBL 814
- 816+LBL 815
- 817+LBL 816
- 818+LBL 817
- 819+LBL 818
- 8

VB41CX Quick Reference Guide .....	1
Introduction .....	3
Keys and Mouse in the Emulation.....	3
The Normal Keyboard .....	4
The User Keyboard.....	5
The ALPHA Keyboard.....	6
The Alarm Catalog Keyboard.....	7
The Stopwatch Keyboard .....	7
The Text Editor Keyboard.....	7
How to Execute Functions (ALPHA Execution) .....	8
Function Set.....	8
Display Features .....	29
Organisation of Memory and Configuration .....	30
Storing and Executing Programs .....	33
Time and Date Formats .....	34
The Catalogues .....	35
The Flags and their Status .....	36
Error Messages .....	37

Copyright 2002,2004,2006,2011,2012 Ron van Tilburg.

No representations about accuracy or fitness for purpose are intended by the author – this is use-at-your-own-risk-ware.

Some materials have been drawn from HP and other publications. Their respective copyrights are hereby acknowledged.

## Introduction

The VB41CX is an emulation (strictly speaking - a black-box simulation) of the Hewlett-Packard HP-41CX calculator, first produced in 1979. It is arguably the last computer produced that was within the capability of a user to completely understand and operate. Highly reliable and brilliantly styled and engineered, it is a pleasure to use.

This emulation, written entirely in VB.NET 2.0 (©Microsoft Corp), is quite complete. While no CPU emulation is used at all, the behavioural features of the machine are well represented and it works very like the real thing – at keystroke and user program execution level as well as its internal data organisation. It cannot however run HP microcode.

Below follows a quick reference guide to this simulation. If you really need to understand at a low level how a RPN scientific calculator works refer to the Web using HP-41CX as a search reference. There are many sites. One, [www.hp41.org](http://www.hp41.org) (by Warren Furlow) is a site entirely dedicated to this machine and has ample material to learn from.

The simulation also (natively) supports the *Synthetic Programming* instructions found by various enthusiasts over the years.

Ron van Tilburg (WEB: [Website](http://Website), email: [rivit@tpg.com.au](mailto:rivit@tpg.com.au))

## Keys and Mouse in the Emulation

The simulation reacts to Mouse or Keyboard for input. When the mouse is used this is exactly as pressing keys on the original HP-41CX. The keyboard reacts as far as possible the same way.

Each PC letter key corresponds to the Letter on the simulator's buttons eg. M=RCL. SHIFT on the PC keyboard acts only as SHIFT on the HP-41CX i.e. combination sequences are not supported. The numeric keypad is supported if the Num Lock is engaged. Enter works as ENTER.

Special Function keys are:

F1 ON/OFF	F2 USER	F3 PRGM	F4 ALPHA
F5 LBL	F6 RTN	F7 CAT	F8 ASN
F9 GTO	F10 BST	F11 SST	F12 R/S

XEQ may be invoked with either ` (accent) or K, ALPHA on/off with ' (single quote)  
Shift ALPHA invokes or cancels HEX mode in ALPHA MODE

## The Normal Keyboard

### Toggle Keys

### Primary Function

### Alternate Function

The Tan Label

### Shift Key

Press first to carry out Alternate Function

### CLx Clear X register

### Back Arrow

backspaces and erases one character at a time (if entry has not been terminated)

### XEQ Execute

Used to execute functions and programs not assigned to keys. See Page 30 in this guide



## The User Keyboard

### USER

Activates the USER keyboard

### Assigning a Function or Global Label to a Key

1. Press **ASN** – ALPHA lights up
2. Enter the Function name or Global Label
3. Press **ALPHA**
4. Press the key to which you want the function assigned. (To restore a key to its Normal Function skip step 2.)

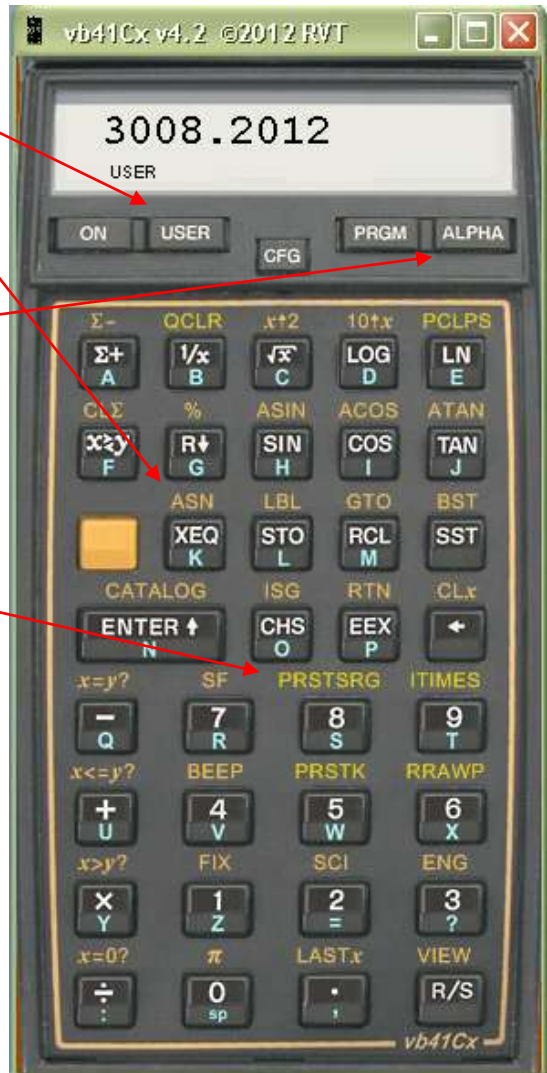
The keyboard label changes on shifted keys to yellow

### Executing a USER function

1. Make sure the **USER** keyboard is active
2. Press the redefined Key
3. Any key not redefined retains its normal function (except the top two rows).

### Local Label Searching

If a key in the top two rows (or shifted top row) is *not* assigned, the VB41CX will perform a local label search if one of the keys is pressed. If a matching Local Label (LBL A thru LBL J or LBL a thru LBL e) is *not* found in the current program, then the Normal function of the key will be executed.



## The ALPHA Keyboard

### Primary Function

The Cyan Letters

### Alternate Function

The Tan Letters

### Shift Key

Press **Shift** first to carry out an alternate function

### Append

Press **⏮** first to have the following ALPHA entry be appended to (rather than overwrite) the previous ALPHA entry

### ALPHA

Activates and deactivates the ALPHA Keyboard

### ASTO Store From ALPHA

Stores the leftmost 6 characters of the ALPHA register into the specified Register

### ARCL Recall Into ALPHA

Recalls the contents of the specified register and appends them to the ALPHA Register

### CLA Clear ALPHA Register

**BackArrow** Backspaces and erases one character at a time (if entry has not been terminated).

### AVIEW View ALPHA Register

Used primarily as a program instruction to display the ALPHA register during a running program



## **The Alarm Catalog Keyboard**

## **The Stopwatch Keyboard**

## **The Text Editor Keyboard**

As of Version 4.2 of VB41CX these keyboards are (still) not implemented.

## How to Execute Functions (ALPHA Execution)

If a function has its own key (whether on the Normal Keyboard or the User Keyboard), you can perform its operation by pressing that key – for example SIN – or by pressing the shift key and then that key – as for **ASIN** (Remember to supply any necessary numbers or labels first).

If a function does not appear on the keyboard – such as TIME – you can perform it using either ALPHA Execution or a USER defined key on the User keyboard. How to assign a function to User Keys is shown on page 3 of this guide. Alpha Execution is shown below:

1. Press XEQ
2. Press ALPHA to activate the Alpha keyboard.
3. Spell out the ALPHA name of the desired function, or the global label of the desired program
4. Press ALPHA to deactivate the Alpha Keyboard and end the procedure.

If the function needs a parameter, it will cue for it with the \_ input cue.

The Function or Program is then executed

Hex input can be started in ALPHA mode by **SHIFT ALPHA** and ended with **SHIFT ALPHA** if an even number of hex digits exists. This input is then converted to characters.

## Function Set

This is an alphabetical list of the VB41CX functions, including brief definitions. For a more detailed summary of these functions, refer to a HP-41CX owner's manual. The VB41CX emulates that machine. Not all functions have been included, but the emulation is quite complete.

Note that you usually supply any needed operands *before* you execute the function (the operator). The exceptions are the *parameter functions*, which cue you for information *after* you execute the function. Parameter functions are shown below with their primary parameter cues, such as **ARCL** nn.

Function names printed in **Blue** are Alpha names and use Alpha execution or User Keyboard execution. Function names in Black or **Gold** are keyboard names, and have keys for execution on the normal keyboards. Names in **Maroon** are recognised in VB41CX but are as yet unimplemented. Enhancements are given in **Green**. New functions by me are in **Pink**.



Function	Definition
←	<i>Back Arrow</i> . Deletion
⏏	<i>Append to Alpha Register</i>
+ (+)	<i>Add y to x</i>
- (-)	<i>Subtract x from y</i>
* (*)	<i>Multiply y by x</i>
/ (÷)	<i>Divide y by x</i>
1/x (1/×)	<i>Reciprocal x</i>
10↑x (10↑×)	<i>Common Exponential x</i>
ABS	<i>Absolute Value x</i>
ACOS	<i>Arc Cosine x</i>
ADATE	<i>Alpha Date</i> Append date to Alpha register
ADV	<i>Advance printer paper</i>
ALENG	<i>Alpha Length.</i> The nr of characters in the Alpha Register
ALMCAT	<i>Alarm Catalog</i>
ALMNOW	<i>Alarm Now.</i> Activate the oldest past-due conditional or control alarm
ALPHA = F4	<i>Alpha keyboard toggle</i>
ANUM	<i>Alpha Number</i> Find the first digit string in Alpha Register and return to x
AOFF	<i>Alpha Keyboard off</i>
AON	<i>Alpha Keyboard on</i>
APPCHR	<i>Append Characters</i> to a record in a text file
APPREC	<i>Append a record.</i> (alpha Register) to a text file
ARCL nn (ARCL nn)	<i>Alpha Recall.</i> Append contents of Register nn to Alpha Register
ARCLREC	<i>Alpha Recall Record</i> Append a text file record to Alpha Register. <b>FS20 and X&gt;0 then only X chars are returned</b>

Function	Definition
AROT	<i>Alpha rotate</i> n places in x
ASHF	<i>Alpha Shift</i> six characters to the left
ASIN	<i>Arc sine</i> x
ASN [name], key (ASN)	Assign function or label to user key
ASROOM	<i>ASCII Room.</i> give the free space in bytes in a text file
ASTO nn (ASTO nn)	<i>Alpha Store</i> Copy the first 6 characters in Alpha Register to the Register nn
ATAN (ATAN)	<i>Arc tangent</i> x
ATIME	<i>Alpha Time.</i> Append time to Alpha Register
ATIME24	<i>Alpha Time 24-hour</i> Append the time to the Alpha Register in CLK24 Format
ATOX	<i>Alpha to X</i> Shift the leftmost character out of the Alpha Register and put its ASCII code in x
AVIEW (AVIEW)	<i>Alpha View</i>
BEEP (BEEP)	<i>Beeper</i> 4 tones
BST (BST) = F11	<i>Back Step</i> through Program
CAT n (CATALOG n)	<i>List Catalog</i> n (1 to 7)
CF nn (CF nn)	<i>Clear Flag</i> nn (00 to 29)
CHS (CHS)	<i>Change Sign</i> of x
CLA (CLA)	<i>Clear Alpha Register</i>
CLALMA	<i>Clear Alarm by Alpha.</i> Clear alarm whose message matches Alpha register
CLALMX	<i>Clear alarm by x.</i> Clear the xth alarm
CLD	<i>Clear Display</i> of Message
CLFL	<i>Clear file</i> named in Alpha (text or data). If CLEARXMEM is given as name clears all of XMemory
CLK12	<i>Clock 12-hour</i> (format)

Function	Definition
CLK24	<i>Clock 24-hour</i> (format)
CLKEYS	<i>Clear all user keys</i>
CLKT	<i>Clock Time only</i> (format)
CLKTD	<i>Clock Time and Date</i> (Format)
CLOCK (ON)	<i>Display Clock</i>
CLP [label]	<i>Clear program</i> with named label, or if none given the current program
CLRALMS	<i>Clear all alarms</i>
CLRG	<i>Clear all data registers</i>
CLRGX	<i>Clear registers by X</i> (bbb.eeeii) Clear every iith register from Rbbb to Reee
CLΣ (CLΣ)	<i>Clear summations.</i> Clear statistical registers
CLST	<i>Clear Stack</i>
CLx (CLx)	<i>Clear x</i> (the usual display)
COPY label	<i>Copy ROM program</i> specified by label
CORRECT	<i>Set time and adjust accuracy factor</i>
COS (COS)	<i>Cosine x</i>
CRFLAS	<i>Create file ASCII</i> Create a text file of name (Alpha) and size (x)
CRFLD	<i>Create File DATA</i> of given name (ALPHA) and size (x)
D-R	<i>Degrees to Radians</i> conversion x
DATE	<i>Value for the Date to x</i>
DATE+	<i>Date Add</i> Add number of days (in x) to date (in y) for a new date
DDAYS	<i>Delta Days</i> find the number of days between dates in x and y registers
DEC	<i>Decimal</i> Octal to Decimal conversion
DEG	<i>Degrees</i> mode set

Function	Definition
DEL nnn	Delete nnn lines from and including current line
DELCHR	Delete n characters from current text file at current pointer
DELREC	Delete current record in a text file
DMY	Day-month-year format
DOW	Day of the week given date (0=Sun,1=Mon...)
DSE nn	Decrement and Skip if Less than or Equal Given iiii.fffcc in Rnn decrement iiii by cc and skip next line if iiii is less than or equal to fff
ED	Text File Editor
EEX	Enter exponent
EMDIR	Extended Memory Directory (Catalog 4)
EMDIRX	Extended Memory Directory by X. Find the xth files name and size
EMROOM	Extended memory Room. Nr of registers left available in extended memory
END	End of Program
ENG n (ENG n)	Engineering Display Use n+1 digits and $10^{3n}$ exponents. Eng F displays results in HEX
ENTER• (ENTER•)	Separate sequential numbers (bump stack)
E↑X (E↑X)	Natural Exponential x
E↑X-1	Natural Exponential x for arguments close to zero
FACT	Factorial x ( $x < 69$ )
FC? nn	Flag nn clear? if not skip next line
FC?C nn	Flag nn clear? if not skip next line always clear flag
FIX n (FIX n)	Fixed point display with n decimal places FIXix F displays results in HEX
FLSIZE	File size (registers) of named (alpha) file
FRC	Fractional Part of x

Function	Definition
<code>FS? nn (FS? nn)</code>	<i>Flag nn set?</i> if not skip next line
<code>FS?C nn</code>	<i>Flag nn set?</i> if not skip next line always clear flag
<code>GETAS</code>	<i>Get ASCII file</i> copy mass storage text file to extended memory
<code>GETKEY</code>	After 10 sec return key code of key pressed or 0 if none
<code>GETKEYX</code>	<i>Get Key by X.</i> After given nr of seconds (x) return Keycode (Y) and character code (X)
<code>GETP</code>	<i>Get Program</i> Replace last program in memory with program file named in Alpha
<code>GETR</code>	<i>Get all registers</i> from given data file and copy to main memory
<code>GETREC</code>	<i>Get Record</i> from current text file and copy to Alpha Register starting at current pointers. If <code>FS20</code> and <code>X&gt;0</code> then only X chars are returned
<code>GETRX</code>	<i>Get Registers by X</i> (bbb.eee) Copy regs in current data file (starting at pointer) to Rbbb to Ree in main memory
<code>GETSUB</code>	<i>Get Subroutine</i> from named file (Alpha) and copy into main memory
<code>GETX</code>	<i>Get X</i> from the current data file
<code>GRAD</code>	<i>Set Gradians</i> mode
<code>GTO label (GTO label)</code>	<i>Go to Program</i> branch to given label
<code>GTO.nnn</code>	<i>Go To (Dot).</i> go to line nnn in program
<code>GTO..</code>	<i>Go To (Dot Dot)</i> Move to end of program memory and pack
<code>HMS</code>	<i>Hours-minutes-seconds</i> convert from decimal hours
<code>HMS+</code>	<i>Hours-minutes-seconds plus</i> Add degrees or times
<code>HMS-</code>	<i>Hours-minutes-seconds minus</i> Subtract degrees or times

Function	Definition
HR	<i>Hours</i> convert from hours-minutes-seconds
INSCHR	<i>Insert Characters</i> from Alpha Register into text file starting at current pointer
INT	<i>Integer Part</i> of x
ISG nn ( <i>ISG</i> nn)	<i>Increment and Skip if Greater than</i> Given iiiii.ffff in Rnn increment iiiii by cc and skip next line if iiiii is greater than fff
LASTX ( <i>LASTx</i> )	<i>Recall number from Last X register</i>
LBL label ( <i>LBL</i> label)	<i>Program label</i>
LN (LN)	<i>Natural log</i> x
LN1+X	<i>Natural log</i> x for arguments close to 1
LOG (LOG)	<i>Common log</i> x
MDY	<i>Month-Day-Year</i> (format)
MEAN	<i>Means</i> of accumulated x and y values
MOD	y <i>mod</i> x
OCT	<i>Octal</i> Decimal to Octal conversion
ON = F1	<i>Continuous on</i>
P-R	<i>Polar to Rectangular</i> conversion Enter Theta, then r returns x to X, y to Y registers
PACK	<i>Pack</i> program memory
PASN	<i>Programmable Assign</i> see ASN
PCLPS	<i>Programmable Clear Programs</i> Clear program named in alpha and all others to end of program memory

Function	Definition
$\%$ ( $\%$ )	$x$ percent of $y$
$\%CH$	Percent change from $y$ to $x$
$PI$ ( $\pi$ )	Value of $\pi$ to 9 places
$POSA$	Position in Alpha Find the position of string(in $x$ ) in alpha register
$POSFL$	Position in File Pointer value of string (in Alpha) in a text file
$PRGM = F3$	Program Mode toggle
$PROMPT$	Display the contents of alpha and stop the program (allowing data input)
$PSE$	Pause Interrupt program for a second
$PSIZE$	Programmable Size see SIZE
$PURFL$	Purge named file
$QRYXFL\ n$	For working or named file return name in ALPHA and the following in X -1 if no working file, or file doesn't exist N=0 X=Filetype 1=P, 2=D, 3=AS, 9=B N=1 X=Filetype "P", "D", "A", "B" N=2 X=Filetype = Ascii code for P, D, A, B N=3 X=ccc char pointer N=4 X=rrr record pointer N=5 X=filesize N=6 X=Sequence nr in Xmemory N=7 X=ASROOM if ascii file N=8 X=Absolute Register of HDR 1 (name) N=9 X=Absolute register of HDR 2
$R\uparrow$	Roll Up stack
$R-D$	Radians to Degrees conversion
$R-P$ ( $R-P$ )	Rectangular to Polar conversion. Enter $y$ , then $x$ returns $r$ in X, $\theta$ in Y
$R/S = F12$	Run/Stop program
$RAD$	Radians Mode
$RCL\ nn$ ( $RCL\ nn$ )	Recall (copy) value from Rnn
$RCLAF$	Recall accuracy factor for clock
$RCLALM$	Recall alarm parameters for alarm $n$

RCLFLAG	<i>Recall flags</i> status for flags 00-43
Function	Definition
RCLPT	<i>Recall pointer</i> value for current file
RCLPTA	<i>Recall pointer</i> value for current file named in Alpha
RCLSW	<i>Recall stopwatch</i> time in HH.MMSSTT
RCLSWM	<i>Recall stopwatch</i> time in Milliseconds
RDN (R↓)	<i>Roll down</i> stack
REGMOVE	<i>Register move</i> Given sss.dddnnn copy nnn registers from Rsss on to Rddd on
REGSWAP	<i>Register swap</i> Given sss.dddnnn swap nnn registers from Rsss on to Rddd on
RESZFL	<i>Resize File</i> (text or data) as specified
RND	<i>Round</i> (to current display digits)
RTN (RTN)	<i>Return</i> program flow from subroutine to main program
RUNSW	<i>Run Stopwatch</i>
SAVEAS	<i>Save ASCII file</i> to mass storage file
SAVEP	<i>Save program</i> to program file named (Alpha)
SAVER	<i>Save all registers</i> to given data file
SAVERX	<i>Save registers by X</i> (bbb.eee) to given data file copy Rbbb thru Reee to current data file
SAVEX	<i>Save X</i> value to current data file
SCI n (SCI n)	<i>Scientific notation</i> with n decimal places SCI F displays results in HEX
SDEV	<i>Standard Deviations</i> of accumulated x and y values
SEEKPT	<i>Seek Pointer</i> Set given pointer value for current text or data file
SEEKPTA	<i>Seek Pointer</i> Set given pointer value for named (Alpha) text or data file
SETAF	<i>Set Accuracy factor</i> for clock
SETDATE	<i>Set date</i> of Clock
SETIME	<i>Set time</i> of clock



Function	Definition
SETSW	<i>Set stopwatch</i> starting time (always set to 0.00)
SF nn (SF nn)	<i>Set flag</i> nn (00 to 29)
$\Sigma+$ ( $\Sigma+$ )	<i>Summation Plus</i> Add data values (x,y) to statistical accumulation
$\Sigma-$ ( $\Sigma-$ )	<i>Summation Minus</i> Delete data values (x,y) from statistical accumulation
$\Sigma$ REG nn	<i>Statistic Registers</i> set to Rnn through Rnn+5
$\Sigma$ REG?	Find number of first statistics register
SIGN	1 or -1 for numbers, 0 for non numbers, +1 for zero
SIN (SIN)	<i>Sine</i> x
SIZE nnn	Allocates nnn registers of storage to data
SIZE?	Nr of registers allocated to storage of data
SQRT ( $\sqrt{x}$ )	<i>Square Root of</i> x
SST (SST) = F11	<i>Single step</i> to next program line
ST+ nn (STO + nn)	<i>Store Plus</i> Rnn=Rnn+x
ST- nn (STO - nn)	<i>Store Minus</i> Rnn=Rnn-x
ST* nn (STO * nn)	<i>Store Times</i> Rnn=Rnn*x
ST/ nn (STO ÷ nn)	<i>Store Divide</i> Rnn=Rnn/x
STO nn (STO nn)	<i>Store</i> copy x into Rnn
STOFLAG	<i>Restore flag</i> status of flags 00-43 from X or restore status of flags bb thru ee given bb.ee in x, and flag data in Y
STOP (R/S)	<i>Stop</i> a running program
STOPSW	<i>Stop Stopwatch</i>
SW	<i>Stopwatch mode</i>
SWPT	<i>Stopwatch and Pointers</i> Given sss.rrr activate Stopwatch keyboard and set storage (sss) and recall (rrr) pointers
T+X	<i>Time plus X</i> Adjust time by increment in x

Function	Definition
TAN (TAN)	<i>Tangent x</i>
TIME	value of <i>current time to x</i>
TONE nn	<i>Sound note</i> 00<nn<99
USER = F2	<i>User keyboard toggle</i>
VIEW nn	Display contents of register Rnn
X↑2 (x↑2)	<i>Square x</i>
X=0? (X=0?)	
X≠0?	
X<0?	
X<=0?	
X>0?	
X=Y? (X=Y?)	
X≠Y?	
X<Y?	<i>Conditional.</i> If not true skips next program line
X<=Y? (X<=Y?)	
X>Y? (X>Y?)	
X=NN?	}
X≠NN?	}
X<NN?	<i>Conditional.</i> Uses the contents of Rnn (NN in Y register) for comparison. If not true skips next program line
X<=NN?	}
X>NN?	}
X>=NN?	}
X<> nn	<i>X exchange</i> with Rnn contents
X<>F	<i>X exchange flags</i> (status of flags 00 to 07)
Function	Definitions

<b>X&lt;&gt;Y</b> ( <b>X&lt;&gt;Y</b> )	X exchange Y
<b>XEQ</b> name ( <b>XEQ</b> name)	<i>Execute</i> the given function or label
<b>XTOA</b>	<i>X to Alpha.</i> Convert x (a character code) to character appended to Alpha
<b>ZYXALM</b>	<i>XYZ Alarm set</i> (see Page 32)
<b>Y^X</b> ( <b>Y^X</b> )	<i>y to the x power</i> (Enter y, then x)
<b>XROM29 Printer Functions</b>	<b>Definition</b>
<b>ACA</b>	<i>Accumulate the Alpha</i> register into Printer Buffer
<b>ACCHR</b>	<i>Accumulates character</i> in x (1 to 127) into Printer Buffer
<b>ACCOL</b>	<i>Accumulates column</i> value in x (0 to 127) into Printer Buffer
<b>ACSPEC</b>	<i>Accumulates special character</i> in x into Printer Buffer
<b>ACX</b>	<i>Accumulates x register</i> value into Printer Buffer
<b>ADV</b>	<i>Prints print buffer Right justified</i>
<b>BLDSPEC</b>	<i>Builds special character.</i> Place column print number into x and execute BLDSPEC. Repeat up to 7 times. Accumulate into printer buffer with ACSPEC or store in a data register
<b>LIST</b> nnn	<i>Lists program lines.</i> Set calculator to line in program mode and execute LIST (not programmable)
<b>PRA</b>	<i>Print the contents of Alpha register</i>
<b>`PRAXIS</b>	<i>Prints Axis</i> using these value R00=YMIN, R01=YMAX, R02=nnn, R04=AXIS
<b>PRBUF</b>	<i>Prints print buffer Left justified</i>
<b>PRFLAGS</b>	<i>Prints flags and other status information</i>
<b>PRKEYS</b>	<i>Prints key assignments</i>
<b>PRP</b> [label]	<i>Prints named program.</i> If name not given prints current program. . <b>If FS?20 then produce an enhanced listing in MAN Mode.</b> Not Programmable
<b>Function</b>	<b>Definitions</b>

<code>`PRPLOT</code>	<i>Prints plot of named function.</i> Prompts for NAME, YMIN, YMAX, AXIS, XMIN, XMAX, XINC. Positive XINC specifies increments between values, negative XINC indicates number of increments. Store optional special plotting character in R03
<code>`PRPLOT P</code>	<i>Prints plot of function.</i> Inputs must be stored in the following registers: R00=YMIN, R01=YMAX, . R03=optional plotting character, R04=AXIS, R08=XMIN, R09=XMAX, R10=XINC. R11=NAME
<code>PRREG</code>	<i>Prints all registers from R00</i>
<code>PRREGX</code>	<i>Prints registers by X (bbb.eee). Print from Rbbb to Reee</i>
<code>PRΣ</code>	<i>Prints summation registers</i>
<code>PRSTK</code>	<i>Prints stack</i>
<code>PRX</code>	<i>Prints X</i>
<code>REGPLOT</code>	<i>Plots single line with data stored in the following registers: X=value, R00=YMIN, R01=YMAX, R02=nnn.aaa where nnn is width (max 168) and aaa is the column of the axis. PRAXIS automatically stores nnn.aaa in R02</i>
<code>SKPCHR</code>	<i>Skips characters in the print buffer</i>
<code>SKPCOL</code>	<i>Skips columns in the print buffer</i>
<code>STKPLT</code>	<i>Plots single line with data stored in the following registers: T=value, Z=YMIN, Y=YMAX, X=nnn.aaa where nnn is width (max 168) and aaa is the column of the axis.</i>
<code>FMT</code>	<i>Format an output value by x and y</i>
<code>DELAY</code>	<i>?</i>
<code>MAN</code>	<i>Set Manual Printing Mode</i>
<code>MAPOFF</code>	<i>?</i>
<code>MAPON</code>	<i>?</i>
<code>NORM</code>	<i>Set NORM printing Mode</i>
<code>PRTOFF</code>	<i>Printer Off</i>
<code>PRTON</code>	<i>Printer ON</i>
<code>RESETP</code>	<i>Reset Printer</i>
<b>Function</b>	<b>Definitions</b>

STARTU	?
STOPU	?
TESTP	<i>Test Printer</i>
TRACE	<i>Set TRACE print Mode</i>
<b>RVT Extensions</b>	<b>Definitions</b>
ACCOLA	<i>Accumulate the ALPHA register as column codes</i>
ACINK	<i>Set ink colour 0-15 in x 0=Black, 1=Blue, 2=Green, 3=Red, 4=Yellow, 5=Cyan, 6=Magenta, 7=White, 8=Dk. Grey, 9= Lt. Blue, 10=Lt. Green, 11=Lt. Red, 12=Lt. Yellow, 13=Lt. Cyan, 14=Lt. Magenta, 15 = Lt. Grey</i>
ATOSPEC	<i>Alpha register to a SPEC character</i>
PRCE fnname	<i>Print Catalog Entry</i> This show all of the flags and parameters, and what a function affects (HELP in disguise)
PRCHRSET	<i>Print entire character set</i>
PRSTSRG	<i>Print Hex dump of status registers</i>
PPRP	<i>Programmable Print Program</i> , name in ALPHA. If ALPHA is * then all programs in memory are listed. If FS20 then produce an enhanced listing in MAN Mode
PRRIP	<i>Dump the printer paper strip into a file, and RIP paper</i> zPRINTSTRIP_VB41c.txt – not all characters are printable and special formatted columns or characters don't work
PRHEX	<i>Print the value in X in Hex Mode</i>
ACXH	<i>Accumulate X in HEX (21 characters)</i>
ACCHRH	<i>Accumulate Char 0,,255) in Hex (2 chars)</i>
ACYX	<i>Accumulate Y by X</i> where X is width. If X<0 then pad with spaces, if x>0 pad with Zeros
`PRXFL	<i>Prints named or working extended memory file in a special format</i>
`MKSPEC	<i>Enables construction of Spec Printing Characters in a simple row-wise way</i>
<b>XROM30 Card Reader</b>	<b>CARD = a disk file of .RAW or .DATA is expected</b>

Functions	Definition
<a href="#">MRG</a>	<i>Merge</i> a ( .RAW) program into the current program after the current line. If at END before this END
<a href="#">RDTA</a>	<i>Read data cards</i> into all data registers ( .DTA File)
<a href="#">RDTAX</a>	<i>Read data cards by X</i> (bbb.eee). Read Rbbb to Reece
<a href="#">RSUB</a>	<i>Read</i> a ( .RAW) <i>subroutine</i> in after the last program in memory if not in last, else add as last program
<a href="#">VER</a>	<i>Verify</i> a .RAW (chksum), .ALL, .STS, .DTA file (mod 7)
<a href="#">WALL</a>	Write the whole machine to a .ALL disk file
<a href="#">WDTA</a>	<i>Write data cards</i> from all data registers .DTA File
<a href="#">WDTAX</a>	<i>Write data cards by X</i> (bbb.eee) Write Rbbb to Reece
<a href="#">WPRV</a>	<i>Write Program in Private Mode</i> as .RAW File
<a href="#">WSTS</a>	<i>Write a Status Register</i> .STS file
<b>HP67 Emulation Functions</b>	<b>Definition</b>
<a href="#">7CLREG</a>	Clear R00-R25
<a href="#">7DSP0</a>	Display 0 decimal places
<a href="#">7DSP1</a>	Display 1 decimal places
<a href="#">7DSP2</a>	Display 2 decimal places
<a href="#">7DSP3</a>	Display 3 decimal places
<a href="#">7DSP4</a>	Display 4 decimal places
<a href="#">7DSP5</a>	Display 5 decimal places
<a href="#">7DSP6</a>	Display 6 decimal places
<a href="#">7DSP7</a>	Display 7 decimal places
<a href="#">7DSP8</a>	Display 8 decimal places
<a href="#">7DSP9</a>	Display 9 decimal places
<a href="#">7DSPI</a>	Display decimal places given in R25
<a href="#">7DSZ</a>	Decrement R25, skip next if equal zero
<b>Function</b>	<b>Definitions</b>

7DSZI	Decrement register pointed to by R25, skip next if = zero
7ENG	Set engineering display mode
7FIX	Set fixed decimal display mode
7GSBI	XEQ the subroutine whose label is in R25
7GTOI	GTO the label in R25
7ISZ	Increment R25, skip next if less than equal zero
7ISZI	Increment register pointed to by R25, skip next if equal zero
7P<>S	Exchange R00-09 with R10-19
7PRREG	<i>Print R00-R25</i>
7PRSTK	<i>Print Stack</i>
7PRTX	<i>Print X</i>
7RCLΣ	<i>Recall Sums</i> SumX and SumY to X,Y
7SCI	Set scientific Display Mode
<b>RVT Extensions</b>	<b>Definitions</b>
7INIT	Establish ΣREG=14 and SIZE 026
RRAWP	<i>Read Raw Program</i> i.e. Program off disk in .raw format
RRAWS	<i>Read Raw SubRoutine</i> Program i.e. Program off disk in .raw format
WRAPW	<i>Write Raw program</i> (named in alpha) as .raw file
RTXTP	<i>Read a program in .TXT form</i> (one that passes the 41uc compiler) into memory as a program
RTXTS	<i>Read a subroutine program in .TXT form</i> (one that passes the 41uc compiler) into memory as a program
RALL	Read a .ALL file produced with WALL
RSTS	Read a .STS file produced with WSTS
RAS	Read a TXT file into an ASCII XMEM file
WAS	Write a TXT file from an XMEM file

<b>XROM28 Mass Storage Functions</b>	<b>Definition</b>
CREATE	Create a file
DIR	List Directory
NEWM	New volume
PURGE	Purge (remove) a file
READA	Read ALL file
READK	Read Key assignment file
READP	Read Program
READR	Read data registers
READRX	Read Data by x
READS	Read Status File
READSUB	Read a Program as subroutine
RENAME	Rename a file
SEC	Secure (protect) a file
SEEKR	Position volume by ordinal
UNSEC	Unsecure (unprotect) file
VERIFY	Verify volume
WRTA	Write an ALL file
WRTK	Write a KEYS file
WRTP	Write a Program file
WRTPV	Write Program as Private file
WRTR	Write Registers Data File
WRTRX	Write Registers by X
WRTS	Write STATUS file
ZERO	Clear a file contents
-CTRL_FNS AUTOIO, FINDID INA, IND, INSTAT LISTEN, LOCAL, MANIO	Recognised but not Implemented



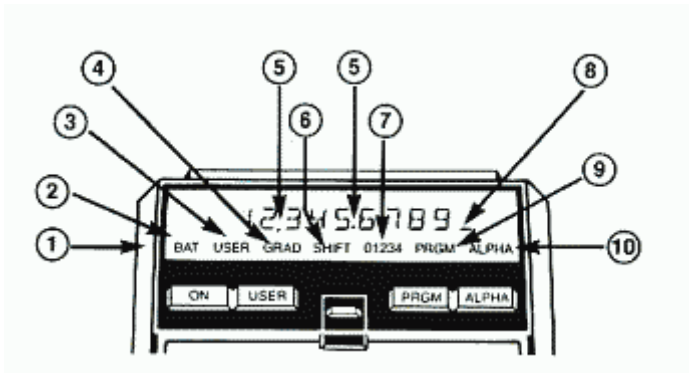
OUTA, PWRDN, PWRUP REMOTE, SELECT STOPIO, TRIGGER	
<b>EXT MS FUNCTIONS</b>	<b>Definitions</b>
PRDIR	<i>Print Directory</i>
COPYFL	<i>Copy a file</i> from one to another
DIRX	Return a file details given file nr. in X
FLENG	<i>File length</i>
FLTYPE	<i>File type</i>
SELM	Select a mass volume (...)
CVRAW	<i>Convert raw</i> files to a volume (RVT extension)
<b>XROM 27 RVT PAGE Q</b>	A Page Device for Graphics Input and Output
QBND\$	Return the <i>Bounds</i> of the current Graphics Page , width in X and Height in Y
QCLR	<i>Clear</i> the current Output Page
QMODE	Set the <i>Mode</i> of the Device 0= Strip Graphics – a copy of printer output 1= Browser Mode 2= Full Page Graphics 3= Scanner Mode
QPOS	Set current cursor <i>position</i> as X=Line,Y=Strip in Mode 0 or X and Y in Mode 2
QPOS?	Return current cursor <i>position</i> as X=Line,Y=Strip in Mode 0 or X and Y in Mode 2, and PageNr in Z
QPRINT	<i>Print</i> the active page on a PC printer
QRESET	<i>Reset</i> to defaults Mode 0 = StripGfx, 800x1100 pixels
QSCROLL	<i>Scroll</i> to Pixel position X,Y
QSEL	<i>Select</i> Page given in X
QSIZE	Set <i>Pagesize</i> in Pixels Width in X, Height in Y. Takes effect on Next New Page
QSIZE?	Return <i>Pagesize</i> Width in X, Page Height in Y

QSKPP	<i>Skip</i> from current to next <i>page</i> Modes 0,2
QSKPS	<i>Skip</i> from Current <i>Strip</i> to Next Strip (or Page if last) Mode 0
QSNAP	<i>Snapshot the current Printpage to Disk</i> and increment pagenn. Pages saved to zPrintPagenn.gif or \Name.gif if name given in ALPHA
QGETXY	<i>Get</i> a Mouse click <i>position</i> on the current page into X,Y
QPXL?	Returns the <i>pixel run</i> at X,Y, and positions to next different pixel horizontally from Left to right . If row finished CF17 else SF17 Stack on exit: X Pen number best matching Y The number of pixels from x,y to new X,Y - Z New (next x) T New (next y)
BCODEP	Create the <i>Barcode</i> for a <i>Program</i> named in ALPHA
`SCANP	<i>Scan Barcode</i> from an image of HP41Barcode, and add that program at the end of <i>Program</i> memory Uses Reg00-20 and extended memory Prompts are :  Get BC Page, R/S Using the browse button ... load the image of the HP Program barcode you wish to read. Position to the top row of barcode. Zoom up a couple of sizes  Get TL Row1, R/S Click on the top left pixel of the 1 <sup>st</sup> row of barcode  Get BR Row1, R/S Click on the bottom right pixel of the 1 <sup>st</sup> row of barcode. It is positioned near there for you This allows calculation of the height of barcode  Get TL Row2, R/S Click on the top left pixel of the 2nd row of barcode. It is positioned near there for you. This allows calculation of the skip between rows  Get BR Last Row, R/S

	<p>Find and click on the Bottom Right pixel of the last row of barcode. This allows calculation of nr of rows.</p> <p>The program then scans the rows attempting to deduce the relative widths of lines, converts to bit widths and then rolls up into program instructions. If successful it loads the program at the end of program memory.</p> <p>If it fails this is usually because there are stray pixels between black stripes in the code, confusing the scanner. Cleanup the barcode, checking particularly around the middle of the bars, and try again.</p>
<b>XROM27 RVT FCNS</b>	Various tools for exploring the HP41
RROM RROMX RROMR	<p><i>Read</i> an entire V41 (by Warren Furlow) .ROM file into program memory. This can adversely affect catalogue, program listings and execution. This is intended for research and conversion activities. Once you code the VB module corresponding to the ROM it is possible to save the .COD file for that ROM with WRCOD. RROMX will attempt to parse the number of ROMs in X at once, (after prompting for the ROM names). RROMR reads one file without parsing it. Not programmable.</p>
WRCOD nn	Make and <i>Write a ROM</i> .COD file of programs corresponding to XROM nn. Not Programmable. This will only succeed if the Module VB code exists and is loaded
WRCODA nn A	Make and <i>Write an autoconfigurable ROM</i> .COD file of programs corresponding for user determined XROM nn. In X and its name in ALPHA. Not Programmable. This creates a .COD file of the given name and number from the programs shown in CAT 1. After creation copy the file to \Statefiles, and modify AutoXROM.txt to include it at next start-up.
XEQA	Indirectly execute the program named in ALPHA
ASNA nn	Assign the ALPHA string (formatted as a correct Register assignment) or 0, into Assignment Register nn 0<=nn<=31
RCLABS nn	RCL the contents of the absolute register in Rnn
STOABS nn	Store X into the Absolute register pointed at in Rnn
X<>ABS nn	Exchange X with contents of the Absolute register pointed

	at in Rnn
DERR	Display an error number nnmm given in X nn=XROM nr, mm = Messagenr
NOT	X = NOT X (all bytes of X)
AND nn	X = X AND Rnn (all bytes of X and Rnn)
OR nn	X = X OR Rnn (all bytes of X and Rnn)
XOR nn	X = X XOR Rnn (all bytes of X and Rnn)
H-D	Convert Hexadecimal (bytes of register X) to its decimal value
D-H	Convert integer decimal value of X to Hexadecimal
DSPH	Display in Hexadecimal Mode
PUSHSTK POPSTK	PUSH Stack to a temporary store and restore with POP PUSH X, Y, Z, T -> tmp -> POP L, Y, Z, T
PUSHA POPA	PUSH ALPHA to a temporary store and restore with POP PUSH M, N, O, P -> tmp -> POP M, N, O, P Up to 12 pushes can be accommodated at one time. The PUSH/POP pair can bracket a function in user code to give the impression of a function i.e. X :=< f(x) Last x := X
CRFLB	Create a XMEM Byte Buffer File
GETB	Get a byte from Buffer File (into X)
PUTA	Save Alpha as By String into B File
PUTB	Put Byte in X (0-255) into BFile

## Display Features



1. **Display Annunciators**
2. **Low Power Condition**
3. **User Keyboard Active**
4. **Current Angular mode**
5. **Digit Separator and Radix Mark Flag 28 Set**  
CF 28 Reverses them  
CF 29 removes the digit separator
6. **Shift Set**  
to cancel press Shift key again
7. **Flags(s) Set**  
flags 00 thru 04
8. **Input Cue**
9. **Program Mode**  
or running program
10. **Alpha Keyboard active**

The display message MEMORY LOST indicates that the memory has been cleared and reset.

*NOTE: Take particular care with the instructions STO c, ASTO c or X<> c which affect the state of memory. Also the instructions STO d, X<> d can prevent normal execution.*

The program execution indicator ( > the flying goose) appears and moves each time the program encounters a label.

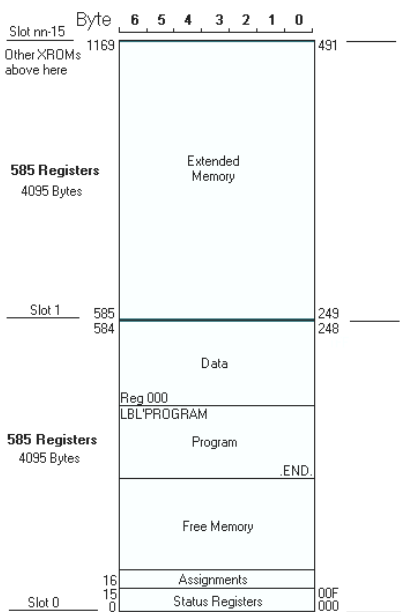
## Organisation of Memory and Configuration

In general the Program Memory is allocated by Module.

### The base VB41CX has:

Main Module 585 Registers Basic (extensible to 4096 Registers)  
 XMemory 585 Registers Basic (extensible to 4096 Registers)  
 XROMs up to 32 modules with up to 585 registers (4095 bytes) each

### VB41 Memory Organisation



v4.2

©2002 Ron van Tilburg

### Status Registers

Name	Byte																Nybble		
	0	1	2	3	4	5	6	7	8	9	A	B	C	D					
e	BITMAP FOR SHIFTED KEY ASSIGNMENTS												LINE NUMBER				F		
d	USER FLAGS 00-29										SYSTEM FLAGS 30-55						E		
c	Σ REG PTR		STK		1		6		9		REG 000 PTR		END. PTR		D				
b	3rd RTN PTR		2nd RTN PTR				1st RTN PTR				PROGRAM PTR				C				
a	6th RTN PTR		5th RTN PTR				4th RTN PTR				3rd RTN				B				
R	BITMAP FOR UNSHIFTED KEY ASSIGNMENTS												Scratch				A		
Q	Scratch																9		
P	DISP FMT		CAT NR		(26)		(25)		ALPHA REGISTER				24		23		22		8
O	21		20		19		18		17		16		15						7
N	14		13		12		11		10		09		08						6
M	07		06		05		04		03		02		01						5
L	LASTx REGISTER																4		
X	X REGISTER																3		
Y	Y REGISTER																2		
Z	Z REGISTER																1		
T	T REGISTER																0		
	Sign				Mantissa								Sign Exponent						

Sign

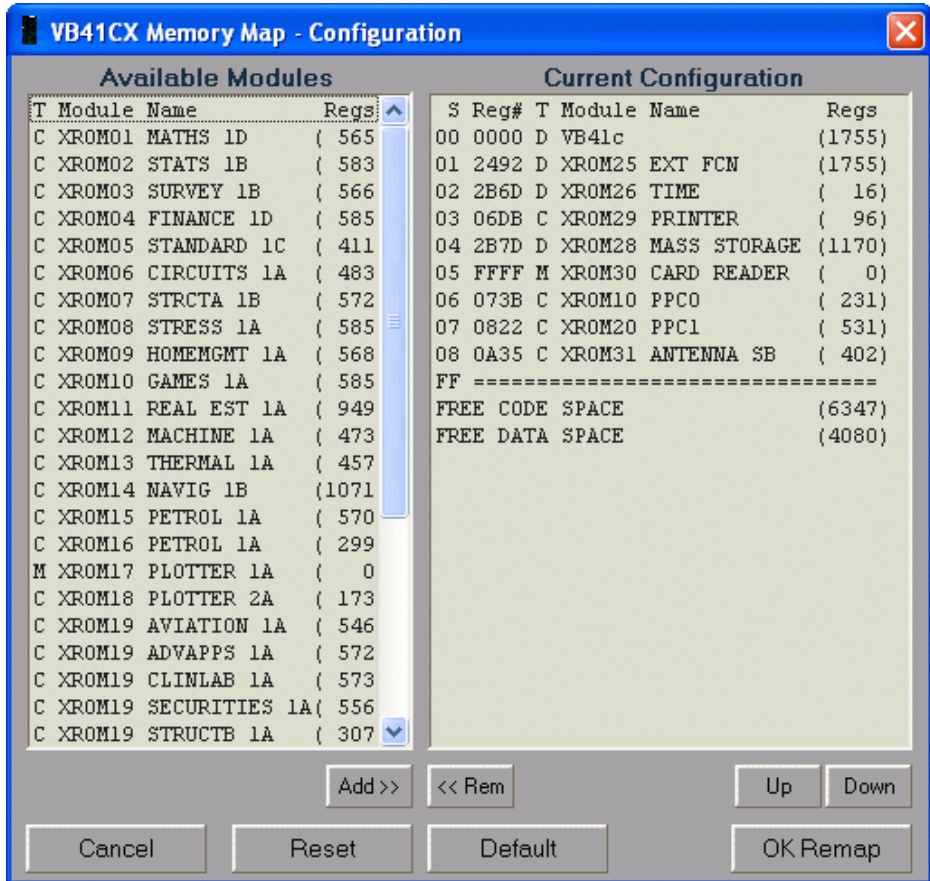
Mantissa

Sign Exponent

### In version 4.2 the default configuration is

Main Memory 1755 Registers  
 XMemory 1755 Registers  
 XROM 25 Extended Functions  
 XROM 30 Printer  
 XROM 27 Card Reader (HP67/97 functions)  
 XROM 28 Mass Storage

**From version 3.2 and later** the default configuration is maximally 64K or 9362 Registers allocated as you wish. Configuration is effected by manipulating the lists shown when then © button is clicked.



The number of registers currently allocated to data storage is given by executing [SIZE?](#).

The number of uncommitted registers (of 7 bytes) still available for use is displayed at the end of Catalog 1 and after pressing [GTO . .](#) in Program Mode.

Whenever Memory is reset (eg. MEMORY LOST = 0 STO c), R00 to R99 are allocated to Data storage. This distribution of registers exists until you change it by executing [SIZE](#) nnn (where nnn is the number of registers to be in data storage).

**Extended Memory** is manipulated in the form of Files which are exchanged with main memory through the so-called Extended functions. Programs and Data can be stored and retrieved from extended memory.

The number of registers still available in extended memory is displayed by [EMROOM](#) and at the end of [CATALOG 4](#).

**XROMS** are conceptually extensions of the main memory containing executable only programs and/or data, and machine code functions. These can significantly extend the utility of the machine as they do not use main storage. The VB41CX executable can contain up to 32 XROMS. These are programmable by a VB programmer (subject to design convention), or creatable as a user code module. (.COD Files)

From version 3.3 a user can make an auto-configuring ROM from the programs in [CAT 1](#). The [WRCODA](#) function (with the name in ALPHA) creates the .COD file and this should be copied to \Statefiles and the AUTOXROM.txt file updated so it will load at next startup

**Mass Storage** is implemented as a flat PC file (.msv) of 228K which can contain up to 500 files. The file consists of a directory and storage. Select (or make a new Volume ) with the [NEWM](#) function after entering a volume name in ALPHA. [CATALOG 7](#) or [DIR](#) will list the contents of the Mass Storage Volume (MSV). Free space is shown at the end of the listing.

**.RAW Program Files** are loaded through [RRAWP](#) and [RRAWS](#) and other CARDREADER functions. A requester is put up to select which files to load. .DAT Files are similarly treated

**.TXT.Program Files** are loaded through [RTXTP](#) and [RTXTS](#). A requester is put up to select which files to load. TXT programs should conform with the established syntax rules acceptable to HP41ue (©Leo Duran). TXT files can also be read into/saved from an ASCII file with [RAS](#), [WAS](#) functions

**The Printer Emulation** features colour and enhanced listings. Additionally there is a text version of the Printer paper-strip kept between [PRRIPs](#) and a graphical image of printer output (800\*1100 pixels) kept between [PRSNAPs](#). Files zPRINTSTRIP.txt, and zPRINTPAGEnn.gif (8 color gif) are saved. This allows session data to be kept in a convenient form. The PrinterStrip output is not suitable for compiling from.



## Storing and Executing Programs

### To store a program in main memory:

1. Press **PRGM** to activate Program mode
2. Press **GTO..** to pack memory and move to the end of program memory.
3. Key in a Global Label of up to 7 characters (eg. LBL ALPHA ABC ALPHA)
4. Key in each subsequent instruction.  
In effect key in the steps you would take in solving the problem.
5. Optional: Press **GTO..** to automatically add an **END** to the program
6. Press **PRGM** to return to Execution Mode

If you make any mistakes use the Backspace key to delete individual characters and entire lines.

### To execute a program in Main Memory:

1. Make sure Execution Mode is active (no **PRGM** annunciator)
2. Start the program by executing its Global label – by Alpha execution – or by User Key. Program execution starts at that global label.

While the program is running the **PRGM** annunciator will be on. The flying goose > will also move in the display.

Pressing R/S (F12) will either start (from its current line) or stop the current program (if it is running). If a running program stops to prompt for data, for example, you key in the data and then press R/S to continue the program.

To run (and re-run) the current program you can press **RTN** R/S

## Time and Date Formats

### Time Values

The calculator interprets clock time values that you specify according to the following conventions:

#### Time Settings

Setting	Clock Time
0	Midnight
1	1 am
2	2 am
...	...
10	10 am
11	11 am
12	Noon
-1 or 13	1 pm or 13:00
-2 or 14	2 pm or 14:00
...	...
-10 or 22	10 pm or 22:00
-11 or 23	11 pm or 23:00
0	Midnight

Results of clock-time operations ([TIME](#), [RCLALM](#)) are always expressed in a 24-hour format in the X-Register. Midnight is Zero.

### Date Values

By default the VB41CX records dates in MM.DDYYYY format and displays them MM/DD/YYYY (US encoding). To set dates to European or British format execute the [DMY](#) function. Dates will then be recorded as DD.MMYYYY and displayed DD.MM.YYYY. Flag 31 is set by this operation

# The Catalogues

There are seven catalogues (press **CATALOG n**) in the VB41CX :

## Catalog 1: User Programs

A list of all global labels and **END** instructions with the byte count for that program, listed in the order in which they were stored. The Permanent end **.END.** shows the number of unused registers in uncommitted memory (and therefore still available for programming).

## Catalog 2: External functions and Time Functions and Extended Functions

A list of all functions and programs currently available to the calculator from peripheral devices, plug in modules and the time, extended and extended-memory functions. The list of functions is grouped by source (press **ENTER**) to see individual functions.

## Catalog 3: Standard Functions

An alphabetical list of the standard (intrinsic) functions of the VB41CX

## Catalog 4: Extended Memory Directory (also **EMDIR**)

A list of all files in extended memory. It gives filename, file type, and the number of registers in the file. It ends with the number of registers left in extended memory.

## Catalog 5: Alarm Catalog – is not Implemented in VB41CX (shows CAT 3)

## Catalog 6: User Key Assignments

A list of all User Key definitions in order of keycode

## Catalog 7: Mass Storage Directory Listing (also **DIR**)

A listing of all files on the currently select mass storage volume. It gives filename, file type, and the number of registers in the file. It ends with the size of the largest available block left on the mass storage volume.

When you execute **CATALOG n**, the catalog listing begins. You can stop and restart it with **R/S**. With the automating listing stopped, you can step forwards through it with **SST** or back with **BST**, or exit the catalog with **BackArrow**.

Most automatic listings speed up when another key is hit. In **TRACE** mode the printer will list all catalog entries.

## The Flags and their Status

0 = Clear

? = Depends on other conditions

1 = Set

M = Maintained by Continuous memory

Flag Number	Flag Name	Status at Reset, Turn-On
<b>00-10</b>	<b>User Flags</b> <b>You can test and alter these flags</b>	<b>0, M</b>
<b>11-29</b>	<b>Control Flags</b> <b>You can test and alter these flags</b>	
11	Automatic execution	0, 0
12-20	External Device Control	0, 0
21	Printer enable (turns Printer Display on/off)	?, ?
22	Numeric Input took place	0, 0
23	Alpha input took place	0, 0
24	Range Error Ignore	0, 0
25	Error ignore if set will clear if error occurred	0, 0
26	Audio Enable	1, 1
27	User Keyboard	0, M
28	Radix Mark	1, M
29	Digit Separator Mark	1, M
<b>30-55</b>	<b>System Flags</b> <b>You can test but not normally alter these flags</b> <i>(synthetic techniques can, but do with care!!!)</i>	
31	Date format ( <b>DMY</b> , <b>MDY</b> changes this)	0, M
32-35	34,35 Page Device	0, M
36-39	Number of Digits ( <b>FIX</b> , <b>SCI</b> , <b>ENG n</b> change these)	4, M
40-41	Display Mode ( <b>FIX</b> , <b>SCI</b> , <b>ENG n</b> change these)	<b>FIX</b> , M
42	Grads Trig Mode	0, M
43	Radians Trig Mode	0, M
44	ON	1, 1
48	ALPHA Keyboard	0, 0
49	Low Power	0, 0
50	Message in Display	0, 0
55	Printer Connected	1, 1

## Error Messages

Following is a simplified list of each error message. For complete descriptions of the error conditions, refer to a real HP manual or look in the source code. The function that caused the error does not get executed, unless Flag 25 is set, which will be cleared on encountering the error. An error message can be cleared by any keyboard action.

Error Message	Meaning
ALPHA DATA	Nonnumeric data used when numeric expected
CHKSUM ERR	Part of Program File lost
DATA ERROR	Illegal operand (wrong values)
DUP FL	File already exists
END OF FL	Pointer is at end of File
END OF REC	Pointer is at end of Record in an ASCII file
FL NOT FND	Specified file (ALPHA) does not exist
FL SIZE ERR	Invalid file size
FL TYPE ERR	Wrong type of file for requested operation
KEYCODE ERR	Nonassignable key given
MEMORY LOST	A global reset has occurred – all contents in main memory are gone
NAME ERR	Invalid file name, empty when required
NO DRIVE	A MSV Drive is not attached
NONEXISTENT	The register, label or function requested does not exist
NO ROOM	Not enough room in Main, Extended or Mass Storage memory
OUT OF RANGE	The result of an operation cannot be represented, or a number is too large
PRIVATE	Program on Drive or ROM is private (not viewable)
RAM	The label specified already exists in main memory
REC TOO LONG	The required operation would result in a record > 254 characters long
ROM	You cannot modify a program in read only memory