

```

000001 2: CPU.PRG V0.2.0a, 7 августа 2022
000002 Процессор, исполняющий команды ПМК
000003
000004 Авторское право (с) Васильев Илья Владимирович, г. Москва
000005 Это свободная программа, защищённая авторским правом
000006 Распространяется под лицензией GNU GPL v3 или старше
000007
000008 OPTION STRICT 'Все переменные должны быть объявлены
000009 OPTION DEFINIT 'Массивы по умолчанию целые
000010
000011 CONST #Hex$="0123456789abcdef"
000012
000013 VAR flgReplace=0 'Флаг перезаписи
000014
000015 VAR shBody$ 'Мантисса вводимого числа или "" (флаг ввода числа)
000016 VAR shExp$ 'Порядок вводимого числа или "" (флаг ввода порядка)
000017 VAR shSign$, shSignExp 'Знаки мантиссы и порядка вводимого числа
000018 VAR shFixed$ 'Команда, фиксирует запятую (флаг введённой запятой)
000019 VAR shFormat$ 'Признак ввода угла: G/M/S
000020 VAR shExpBack$ 'Старый порядок, со знаком, для изменения ВП или ""
000021
000022
000023 'Считать мантиссу из A$, как число, и перевести её в градусы, если flgLand
000024 DEF GetBody#(A$, flgLand)
000025 VAR I, ID=0
000026 CASE shFormat$
000027 WHEN "M":
000028 VAR rM#=0
000029 I=INSTR(A$, ".")
000030 IF I>=0 THEN ID=VAL(MID$(A$, 0, I)): A$=MID$(A$, I+1)
000031 I=INSTR(A$, ",")
000032 IF I>=0 THEN A$[I]="."
000033 rM#=VAL(A$)
000034 WHILE rM#>=60: DEC rM#, 60: INC ID: WEND 'должно быть меньше 60 минут
000035 IF flgLand THEN shFormat$="G": RETURN ID+rM#/60
000036 RETURN ID+rM#/100
000037 WHEN "S":
000038 VAR iM=0, rS#=0
000039 I=INSTR(A$, ".")
000040 IF I>=0 THEN ID=VAL(MID$(A$, 0, I)): A$=MID$(A$, I+1)
000041 I=INSTR(A$, ",")
000042 IF I>=0 THEN iM=VAL(MID$(A$, 0, I)): A$=MID$(A$, I+1)
000043 rS#=VAL(A$)
000044 WHILE rS#>=60: DEC rS#, 60: INC iM: WEND 'должно быть меньше 60 секунд
000045 WHILE iM>=60: DEC iM, 60: INC ID: WEND 'должно быть меньше 60 минут
000046 IF flgLand THEN shFormat$="G": RETURN ID+(iM*60+rS#)/3600
000047 RETURN ID+(iM*100+rS#)/10000
000048 WHEN "G": I=INSTR(A$, ".")
000049 OTHERWISE: I=INSTR(A$, ",") 'Заменяем десятичную запятую на точку
000050 ENDCASE
000051 IF I>=0 THEN A$[I]="."
000052 RETURN VAL(A$)
000053 END
000054
000055
000056 'Поместить введённое число в RX
000057 'лучше точная реализация в строке, без промежуточной двоичной арифметики
000058 DEF Finalize$
000059 IF shBody$="" THEN
000060 VAR Tmp$=""
000061 IF shExp$="" THEN
000062 Tmp$=DecCheck$(GetBody$(shBody$, 0))
000063 ELSE
000064 E=VAL(shExp$)
000065 IF shSignExp="" THEN E=-E
000066 IF shExpBack$="" THEN INC E, VAL(shExpBack$) 'Учитываем знак порядка
000067 Tmp$=DecCheck$(GetBody$(shBody$, 1)*POW(10, E)) 'Старый порядок до ВП
000068 ENDIF
000069 IF VAR("0:ERROR")="" THEN
000070 IF shSign="" && !IsZero(Tmp$) THEN
000071 CASE shFormat$ 'Ставим отрицательный знак мантиссы
000072 WHEN "G": Tmp$[0]="g"
000073 WHEN "M": Tmp$[0]="m"
000074 WHEN "S": Tmp$[0]="s"
000075 OTHERWISE: Tmp$[0]="-"
000076 ENDCASE
000077 ELSEIF shFormat$="" THEN Tmp$[0]=shFormat$
000078 ENDIF
000079 SetX$ Tmp$
000080 shBody$=""
000081 ENDIF
000082 END
000083
000084
000085
000086
000087 'Разобрать формат G: G, g
000088 COMMON DEF DecryptD A$ OUT strBody$, chExpSign$, strExp$
000089 chExpSign$=A$[15]: strExp$=MID$(A$, 16, 2): VAR I=VAL(strExp$)
000090 IF chExpSign$="" OR I=0 THEN 'Отрицательный порядок
000091 strBody$=A$[1]+MID$(A$, 2, 13)
000092 IF I=0 THEN strExp$="": chExpSign$=""
000093 ELSEIF I>13 THEN
000094 strBody$=A$[1]+MID$(A$, 2, 13)
000095 ELSEIF I=13 THEN
000096 strBody$=MID$(A$, 4, 14)+MID$(A$, 15, 1)
000097 strExp$="": chExpSign$=""
000098 ELSE
000099 strExp$="": chExpSign$=""
000100 strBody$=MID$(A$, 1, I+1)+MID$(A$, I+2, 13-I)
000101 ENDIF
000102 I=LAST(strBody$)
000103 WHILE strBody$[I]="0": DEC I: WEND
000104 strBody$=LEFT$(strBody$, I+1)
000105 END
000106
000107
000108 'Разобрать формат M: G, MMM
000109 COMMON DEF DecryptM A$ OUT strBody$, chExpSign$, strExp$
000110 chExpSign$=A$[15]: strExp$=MID$(A$, 16, 2): VAR I=VAL(strExp$)
000111 IF I=0 THEN 'Нулевой порядок, можно убрать в положительный порядок
000112 IF A$[1]="0" THEN strBody$="" ELSE strBody$=A$[1]+MID$(A$, 2, 13)
000113 strExp$="": chExpSign$=""
000114 IF A$[2]="0" THEN Push strBody$, A$[2]
000115 Push strBody$, A$[3]+MID$(A$, 4, 11)
000116 ELSEIF chExpSign$="" THEN 'Отрицательный порядок
000117 CASE I
000118 WHEN 1: strBody$=A$[1]+A$[2]+MID$(A$, 3, 12)
000119 WHEN 2: strBody$=A$[1]+MID$(A$, 2, 13)
000120 OTHERWISE: strBody$=A$[1]+MID$(A$, 2, 13)
000121 strExp$=RIGHT$("0"+STR$(I-2), 2)
000122 ENDCASE
000123 ELSEIF I>13 THEN
000124 strBody$=A$[1]+MID$(A$, 2, 13)
000125 strExp$="": chExpSign$=""

```

```

124 ENDIF I>13 THEN
125 strBody$=A$[1]+""+MID$(A$,2,13)
126 ELSE
127 strBody$=MID$(A$,1,I+1)+""
128 strExp$="":chExpSign$=""
129 CASE I
130 WHEN 13:
131 PUSH strBody$, "0"
132 WHEN 12:
133 IF A$[14]!="0" THEN PUSH strBody$,A$[14]
134 PUSH strBody$, "0"
135 WHEN 11:
136 IF A$[13]!="0" THEN PUSH strBody$,A$[13]
137 PUSH strBody$,A$[14]+""
138 OTHERWISE:
139 IF A$[I+2]!="0" THEN PUSH strBody$,A$[I+2]
140 PUSH strBody$,A$[I+3]+""+MID$(A$,I+4,11-I)
141 ENDIF
142 I=LAST(strBody$)
143 WHILE strBody$[I]="0":DEC I:WEND
144 strBody$=LEFT$(strBody$,I+1)
145 END
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248

```

Разобрать формат MC: г, MMCCс

```

COMMON DEF DecryptMS A$ OUT strBody$,chExpSign$,strExp$
chExpSign$=A$[16]:strExp$=MID$(A$,16,2):VAR I=VAL(strExp$)
IF I=0 THEN
IF A$[1]!="0" THEN strBody$="" ELSE strBody$=A$[1]+""
strExp$="":chExpSign$=""
IF A$[2]!="0" THEN
PUSH strBody$,A$[2]+A$[3]+""
ELSEIF A$[3]!="0" OR strBody$!="" THEN
PUSH strBody$,A$[3]+""
ENDIF
IF A$[4]!="0" THEN PUSH strBody$,A$[4]
PUSH strBody$,A$[5]+""+MID$(A$,6,9)
ELSEIF chExpSign$!="" THEN "Отрицательный порядок"
CASE I
WHEN 1:
strBody$=A$[1]+A$[2]+""
IF A$[3]!="0" THEN PUSH strBody$,A$[3]
PUSH strBody$,A$[4]+""+MID$(A$,5,10)
strExp$="":chExpSign$=""
strBody$=A$[1]+""
IF A$[2]!="0" THEN PUSH strBody$,A$[2]
PUSH strBody$,A$[3]+""+MID$(A$,4,11)
strExp$="":chExpSign$=""
strBody$=A$[1]+A$[2]+""+MID$(A$,3,12)
strExp$="":chExpSign$=""
strBody$=A$[1]+""+MID$(A$,2,13)
strExp$="":chExpSign$=""
strBody$=A$[1]+""+MID$(A$,2,13)
strExp$=RIGHT$("0"+STR$(I-4),2)
ENDCASE
ELSEIF I>13 THEN
strBody$=A$[1]+""+MID$(A$,2,13)
ELSE
strBody$=MID$(A$,1,I+1)+""
strExp$="":chExpSign$=""
CASE I
WHEN 13:
PUSH strBody$, "0"
WHEN 12:
IF A$[14]!="0" THEN PUSH strBody$,A$[14]
PUSH strBody$, "0"
WHEN 11:
IF A$[13]!="0" THEN PUSH strBody$,A$[13]
PUSH strBody$,A$[14]+""
WHEN 10:
IF A$[12]!="0" THEN PUSH strBody$,A$[12]
PUSH strBody$,A$[13]+""
IF A$[14]!="0" THEN PUSH strBody$,A$[14]
PUSH strBody$, "0"
WHEN 9:
IF A$[11]!="0" THEN PUSH strBody$,A$[11]
PUSH strBody$,A$[12]+""
IF A$[13]!="0" THEN PUSH strBody$,A$[13]
PUSH strBody$,A$[14]+""
OTHERWISE:
IF A$[I+2]!="0" THEN PUSH strBody$,A$[I+2]
PUSH strBody$,A$[I+3]+""
IF A$[I+4]!="0" THEN PUSH strBody$,A$[I+4]
PUSH strBody$,A$[I+5]+""+MID$(A$,I+6,9-I)
ENDIF
I=LAST(strBody$)
WHILE strBody$[I]="0":DEC I:WEND
strBody$=LEFT$(strBody$,I+1)
END

```

Дать мантиссу, порядок десятичного числа, а также их знаки

Также разобрать гексы, строки и углы

+ строки возвращаются в кодировке МК-261

```

COMMON DEF DecryptDec A$ OUT chSign$,strBody$,chExpSign$,strExp$
chSign$=A$[0]
CASE chSign$
WHEN "$":
strBody$=MID$(A$,1)
WHEN "#":
strBody$=MID$(A$,1,4)+""+MID$(A$,5,4)
strExp$=""
WHEN "G":
WHEN "g":
DecryptD A$ OUT strBody$,chExpSign$,strExp$
WHEN "M":
WHEN "m":
DecryptM A$ OUT strBody$,chExpSign$,strExp$
WHEN "S":
WHEN "s":
DecryptMS A$ OUT strBody$,chExpSign$,strExp$
WHEN "-":
chExpSign$=A$[15]
strExp$=MID$(A$,16,2):VAR I=VAL(strExp$)
IF chExpSign$!="" THEN
IF A$[1]!="0" THEN strBody$="" ELSE strBody$=A$[1]+""
ELSEIF I>13 THEN
strBody$=A$[1]+""+MID$(A$,2,13)
ELSE
strExp$="":chExpSign$=""
strBody$=MID$(A$,1,I+1)+""+MID$(A$,I+2,13-I)
ENDIF
I=LAST(strBody$)
WHILE strBody$[I]="0":DEC I:WEND
strBody$=LEFT$(strBody$,I+1)
ENDCASE

```



```

241 strBody$=LEFT$(strBody$,l+1)
242 ENDCASE
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

```

390 DecryptRX OUT chSign$,strBody$,chExpSign$,strExp$
391 shBody$=strBody$
392 shSign=(chSign$=="-")
393 IF strExp$=="00" THEN shExpBack$="" \
394 ELSE shExpBack$=chExpSign$+strExp$
395 ENDF
396 shExp$="00"
397 shSignExp=0
398 ELSEIF shExp$!="" THEN
399 Finalize
400 IF VAR("0:ERROR")==0 THEN
401 DecryptRX OUT chSign$,strBody$,chExpSign$,strExp$
402 shBody$=strBody$
403 shSign=(chSign$=="-")
404 shExp$="00"
405 shSignExp=0
406 IF strExp$=="00" THEN shExpBack$="" \
407 ELSE shExpBack$=chExpSign$+strExp$
408 ENDF
409 ELSE 'вп после ввода мантииссы
410 shExp$="00"
411 shSignExp=0
412 shExpBack$=""
413 ENDF
414 shFixed=1:shFormat$=""
415 flgReplace=1
416 END
417
418 'коп 1f -- ГМС
419 COMMON DEF DoDMS
420 IF shBody$="" THEN
421 IF flgReplace==0 THEN StackUpFast
422 shBody$="0"
423 shSign=0
424 shFixed=1
425 shFormat$="G"
426 shExp$=""
427 shExpBack$=""
428 shSignExp=0
429 ELSEIF shExp$="" THEN
430 VAR I=LAST(shBody$)
431 CASE shFormat$
432 WHEN " ":
433 IF shBody$[I]=="." THEN
434 shBody$[I]="°":shFormat$="G":shFixed=1
435 ELSE
436 I=INSTR(shBody$,".")
437 IF I>0 THEN shBody$[I]="°":shFormat$="G":shFixed=1
438 ENDF
439 WHEN "G":
440 IF shBody$[I]=="°" THEN shBody$[I]="/" ELSE PUSH shBody$,"/"
441 shFormat$="M":shFixed=1
442 WHEN "M":
443 IF shBody$[I]="/" THEN shBody$[I]="" ELSE PUSH shBody$,"/"
444 shFormat$="S":shFixed=1
445 WHEN "S":
446 IF shBody$[I]="" THEN
447 IF INSTR(shBody$,"/")=-1 AND INSTR(shBody$,"°")=-1 THEN
448 shBody$[I]=",":shFormat$="":shFixed=0
449 ELSE
450 VAR T$
451 IF INSTR(shBody$,"/")>0 THEN
452 shFormat$="M":T$=POP(shBody$)
453 ELSE
454 IF INSTR(shBody$,"°")>0 THEN shFormat$="G":T$=POP(shBody$)
455 ENDF
456 ENDF
457 ENDIF
458 ENDIF
459 ENDIF
460 ENDIF
461 ENDIF
462 ENDIF
463 ENDIF
464 ENDIF
465 ENDIF
466 ENDIF
467 ENDIF
468 ENDIF
469 ENDIF
470 ENDIF
471 ENDIF
472 ENDIF
473 ENDIF
474 ENDIF
475 ENDIF
476 ENDIF
477 ENDIF
478 ENDIF
479 ENDIF
480 ENDIF
481 ENDIF
482 ENDIF
483 ENDIF
484 ENDIF
485 ENDIF
486 ENDIF
487 ENDIF
488 ENDIF
489 ENDIF
490 ENDIF
491 ENDIF
492 ENDIF
493 ENDIF
494 ENDIF
495 ENDIF
496 ENDIF
497 ENDIF
498 ENDIF
499 ENDIF
500 ENDIF
501 ENDIF
502 ENDIF
503 ENDIF
504 ENDIF
505 ENDIF
506 ENDIF
507 ENDIF
508 ENDIF
509 ENDIF
510 ENDIF
511 ENDIF
512 ENDIF
513 ENDIF
514 ENDIF
515 ENDIF
516 ENDIF
517 ENDIF
518 ENDIF
519 ENDIF
520 ENDIF
521 ENDIF
522 ENDIF
523 ENDIF
524 ENDIF
525 ENDIF
526 ENDIF
527 ENDIF
528 ENDIF
529 ENDIF
530 ENDIF
531 ENDIF
532 ENDIF
533 ENDIF
534 ENDIF
535 ENDIF
536 ENDIF
537 ENDIF
538 ENDIF
539 ENDIF
540 ENDIF
541 ENDIF
542 ENDIF
543 ENDIF
544 ENDIF
545 ENDIF
546 ENDIF
547 ENDIF
548 ENDIF
549 ENDIF
550 ENDIF
551 ENDIF
552 ENDIF
553 ENDIF
554 ENDIF
555 ENDIF
556 ENDIF
557 ENDIF
558 ENDIF
559 ENDIF
560 ENDIF
561 ENDIF
562 ENDIF
563 ENDIF
564 ENDIF
565 ENDIF
566 ENDIF
567 ENDIF
568 ENDIF
569 ENDIF
570 ENDIF
571 ENDIF
572 ENDIF
573 ENDIF
574 ENDIF
575 ENDIF
576 ENDIF
577 ENDIF
578 ENDIF
579 ENDIF
580 ENDIF
581 ENDIF
582 ENDIF
583 ENDIF
584 ENDIF
585 ENDIF
586 ENDIF
587 ENDIF
588 ENDIF
589 ENDIF
590 ENDIF
591 ENDIF
592 ENDIF
593 ENDIF
594 ENDIF
595 ENDIF
596 ENDIF
597 ENDIF
598 ENDIF
599 ENDIF
600 ENDIF
601 ENDIF
602 ENDIF
603 ENDIF
604 ENDIF
605 ENDIF
606 ENDIF
607 ENDIF
608 ENDIF
609 ENDIF
610 ENDIF
611 ENDIF
612 ENDIF
613 ENDIF
614 ENDIF
615 ENDIF
616 ENDIF
617 ENDIF
618 ENDIF
619 ENDIF
620 ENDIF
621 ENDIF
622 ENDIF
623 ENDIF
624 ENDIF
625 ENDIF
626 ENDIF
627 ENDIF
628 ENDIF
629 ENDIF
630 ENDIF
631 ENDIF
632 ENDIF
633 ENDIF
634 ENDIF
635 ENDIF
636 ENDIF
637 ENDIF
638 ENDIF
639 ENDIF
640 ENDIF
641 ENDIF
642 ENDIF
643 ENDIF
644 ENDIF
645 ENDIF
646 ENDIF
647 ENDIF
648 ENDIF
649 ENDIF
650 ENDIF
651 ENDIF
652 ENDIF
653 ENDIF
654 ENDIF
655 ENDIF
656 ENDIF
657 ENDIF
658 ENDIF
659 ENDIF
660 ENDIF
661 ENDIF
662 ENDIF
663 ENDIF
664 ENDIF
665 ENDIF
666 ENDIF
667 ENDIF
668 ENDIF
669 ENDIF
670 ENDIF
671 ENDIF
672 ENDIF
673 ENDIF
674 ENDIF
675 ENDIF
676 ENDIF
677 ENDIF
678 ENDIF
679 ENDIF
680 ENDIF
681 ENDIF
682 ENDIF
683 ENDIF
684 ENDIF
685 ENDIF
686 ENDIF
687 ENDIF
688 ENDIF
689 ENDIF
690 ENDIF
691 ENDIF
692 ENDIF
693 ENDIF
694 ENDIF
695 ENDIF
696 ENDIF
697 ENDIF
698 ENDIF
699 ENDIF
700 ENDIF
701 ENDIF
702 ENDIF
703 ENDIF
704 ENDIF
705 ENDIF
706 ENDIF
707 ENDIF
708 ENDIF
709 ENDIF
710 ENDIF
711 ENDIF
712 ENDIF
713 ENDIF
714 ENDIF
715 ENDIF
716 ENDIF
717 ENDIF
718 ENDIF
719 ENDIF
720 ENDIF
721 ENDIF
722 ENDIF
723 ENDIF
724 ENDIF
725 ENDIF
726 ENDIF
727 ENDIF
728 ENDIF
729 ENDIF
730 ENDIF
731 ENDIF
732 ENDIF
733 ENDIF
734 ENDIF
735 ENDIF
736 ENDIF
737 ENDIF
738 ENDIF
739 ENDIF
740 ENDIF
741 ENDIF
742 ENDIF
743 ENDIF
744 ENDIF
745 ENDIF
746 ENDIF
747 ENDIF
748 ENDIF
749 ENDIF
750 ENDIF
751 ENDIF
752 ENDIF
753 ENDIF
754 ENDIF
755 ENDIF
756 ENDIF
757 ENDIF
758 ENDIF
759 ENDIF
760 ENDIF
761 ENDIF
762 ENDIF
763 ENDIF
764 ENDIF
765 ENDIF
766 ENDIF
767 ENDIF
768 ENDIF
769 ENDIF
770 ENDIF
771 ENDIF
772 ENDIF
773 ENDIF
774 ENDIF
775 ENDIF
776 ENDIF
777 ENDIF
778 ENDIF
779 ENDIF
780 ENDIF
781 ENDIF
782 ENDIF
783 ENDIF
784 ENDIF
785 ENDIF
786 ENDIF
787 ENDIF
788 ENDIF
789 ENDIF
790 ENDIF
791 ENDIF
792 ENDIF
793 ENDIF
794 ENDIF
795 ENDIF
796 ENDIF
797 ENDIF
798 ENDIF
799 ENDIF
800 ENDIF
801 ENDIF
802 ENDIF
803 ENDIF
804 ENDIF
805 ENDIF
806 ENDIF
807 ENDIF
808 ENDIF
809 ENDIF
810 ENDIF
811 ENDIF
812 ENDIF
813 ENDIF
814 ENDIF
815 ENDIF
816 ENDIF
817 ENDIF
818 ENDIF
819 ENDIF
820 ENDIF
821 ENDIF
822 ENDIF
823 ENDIF
824 ENDIF
825 ENDIF
826 ENDIF
827 ENDIF
828 ENDIF
829 ENDIF
830 ENDIF
831 ENDIF
832 ENDIF
833 ENDIF
834 ENDIF
835 ENDIF
836 ENDIF
837 ENDIF
838 ENDIF
839 ENDIF
840 ENDIF
841 ENDIF
842 ENDIF
843 ENDIF
844 ENDIF
845 ENDIF
846 ENDIF
847 ENDIF
848 ENDIF
849 ENDIF
850 ENDIF
851 ENDIF
852 ENDIF
853 ENDIF
854 ENDIF
855 ENDIF
856 ENDIF
857 ENDIF
858 ENDIF
859 ENDIF
860 ENDIF
861 ENDIF
862 ENDIF
863 ENDIF
864 ENDIF
865 ENDIF
866 ENDIF
867 ENDIF
868 ENDIF
869 ENDIF
870 ENDIF
871 ENDIF
872 ENDIF
873 ENDIF
874 ENDIF
875 ENDIF
876 ENDIF
877 ENDIF
878 ENDIF
879 ENDIF
880 ENDIF
881 ENDIF
882 ENDIF
883 ENDIF
884 ENDIF
885 ENDIF
886 ENDIF
887 ENDIF
888 ENDIF
889 ENDIF
890 ENDIF
891 ENDIF
892 ENDIF
893 ENDIF
894 ENDIF
895 ENDIF
896 ENDIF
897 ENDIF
898 ENDIF
899 ENDIF
900 ENDIF
901 ENDIF
902 ENDIF
903 ENDIF
904 ENDIF
905 ENDIF
906 ENDIF
907 ENDIF
908 ENDIF
909 ENDIF
910 ENDIF
911 ENDIF
912 ENDIF
913 ENDIF
914 ENDIF
915 ENDIF
916 ENDIF
917 ENDIF
918 ENDIF
919 ENDIF
920 ENDIF
921 ENDIF
922 ENDIF
923 ENDIF
924 ENDIF
925 ENDIF
926 ENDIF
927 ENDIF
928 ENDIF
929 ENDIF
930 ENDIF
931 ENDIF
932 ENDIF
933 ENDIF
934 ENDIF
935 ENDIF
936 ENDIF
937 ENDIF
938 ENDIF
939 ENDIF
940 ENDIF
941 ENDIF
942 ENDIF
943 ENDIF
944 ENDIF
945 ENDIF
946 ENDIF
947 ENDIF
948 ENDIF
949 ENDIF
950 ENDIF
951 ENDIF
952 ENDIF
953 ENDIF
954 ENDIF
955 ENDIF
956 ENDIF
957 ENDIF
958 ENDIF
959 ENDIF
960 ENDIF
961 ENDIF
962 ENDIF
963 ENDIF
964 ENDIF
965 ENDIF
966 ENDIF
967 ENDIF
968 ENDIF
969 ENDIF
970 ENDIF
971 ENDIF
972 ENDIF
973 ENDIF
974 ENDIF
975 ENDIF
976 ENDIF
977 ENDIF
978 ENDIF
979 ENDIF
980 ENDIF
981 ENDIF
982 ENDIF
983 ENDIF
984 ENDIF
985 ENDIF
986 ENDIF
987 ENDIF
988 ENDIF
989 ENDIF
990 ENDIF
991 ENDIF
992 ENDIF
993 ENDIF
994 ENDIF
995 ENDIF
996 ENDIF
997 ENDIF
998 ENDIF
999 ENDIF
1000 ENDIF

```



```

000493 'Коп fb -- Косвенная запись содержимого регистра X
000494 по содержимому адресных регистров от 0 до ffff
000495 COMMON DEF DoPKM R
000496 Finalize
000497 IF VAR("0:ERROR")==0 THEN KSTO R
000498 flgReplace=0
000499 END
000500
000501
000502 'Коп d0..df -- Косвенный вызов в регистр X
000503 по содержимому адресного регистра R
000504 COMMON DEF DoKRM R
000505 Finalize
000506 IF VAR("0:ERROR")==0 THEN KRCL1 R
000507 flgReplace=0
000508 END
000509
000510
000511 'Коп fd -- Косвенный вызов в регистр X
000512 по содержимому адресных регистров от 0 до ffff
000513 COMMON DEF DoPKRM R
000514 Finalize
000515 IF VAR("0:ERROR")==0 THEN KRCL R
000516 flgReplace=0
000517 END
000518
000519
000520 'Выполнить команду с кодом COP и параметром N
000521 COMMON DEF DO2 COP,N
000522 CASE COP
000523 WHEN &h4: DoM N 'Р П
000524 WHEN &h5: DoM N 'БП
000525 Finalize: VAR("0:PC")=N: flgReplace=0
000526 WHEN &h6: DoRM N 'Р ИП
000527 WHEN &h8: DoRM N 'РК БП
000528 WHEN &hb: DoPKM N 'РК ПП
000529 WHEN &hd: DoPKRM N 'РК ИП
000530 ENDCASE
000531 END
000532
000533 'Выполнить команду с кодом COP
000534 COMMON DEF DO COP
000535 IF COP<10 THEN
000536 Digit COP
000537 ELSEIF COP>=&h40 && COP<=&h4f THEN
000538 DoM COP AND 15 'п0..пf
000539 ELSEIF COP>=&h60 && COP<=&h6f THEN
000540 DoRM COP AND 15 'ип0..ипf
000541 ELSEIF COP>=&hb0 && COP<=&hbf THEN
000542 DoKM COP AND 15 'кп0..кпf
000543 ELSEIF COP>=&hd0 && COP<=&hdf THEN
000544 DoKRM COP AND 15 'кип0..кипf
000545 ELSE
000546 CASE COP
000547 WHEN &h0a: C omma
000548 WHEN &h0b: n n e g a t e
000549 WHEN &h0c: B o d y $ = " : C l e a r X : f l g R e p l a c e = 1
000550 WHEN &h0d: S t a c k U p : f l g R e p l a c e = 0
000551 WHEN &h0e: U n a r y ( " M a t h A d d " ) : f l g R e p l a c e = 0
000552 WHEN &h0f: U n a r y ( " M a t h S u b " ) : f l g R e p l a c e = 0
000553 WHEN &h10: U n a r y ( " M a t h M u l " ) : f l g R e p l a c e = 0
000554 WHEN &h11: U n a r y ( " M a t h D i v " ) : f l g R e p l a c e = 0
000555 WHEN &h12: S t a c k S w a p : f l g R e p l a c e = 0
000556 WHEN &h13: U n a r y ( " M a t h P o w e r 10 " ) : f l g R e p l a c e = 0
000557 WHEN &h14: U n a r y ( " M a t h E x p " ) : f l g R e p l a c e = 0
000558 WHEN &h15: U n a r y ( " M a t h L g " ) : f l g R e p l a c e = 0
000559 WHEN &h16: U n a r y ( " M a t h A c c o s " ) : f l g R e p l a c e = 0
000560 WHEN &h17: U n a r y ( " M a t h A s i n " ) : f l g R e p l a c e = 0
000561 WHEN &h18: U n a r y ( " M a t h A t a n " ) : f l g R e p l a c e = 0
000562 WHEN &h19: U n a r y ( " M a t h C o s " ) : f l g R e p l a c e = 0
000563 WHEN &h1a: U n a r y ( " M a t h S i n " ) : f l g R e p l a c e = 0
000564 WHEN &h1b: U n a r y ( " M a t h T a n " ) : f l g R e p l a c e = 0
000565 WHEN &h1c: U n a r y ( " M a t h C o t " ) : f l g R e p l a c e = 0
000566 WHEN &h1d: U n a r y ( " M a t h S e c " ) : f l g R e p l a c e = 0
000567 WHEN &h1e: U n a r y ( " M a t h C s c " ) : f l g R e p l a c e = 0
000568 WHEN &h1f: D o D M S
000569 WHEN &h20: i z e : L o a d P i : f l g R e p l a c e = 0
000570 WHEN &h21: U n a r y ( " M a t h S q r t " ) : f l g R e p l a c e = 0
000571 WHEN &h22: U n a r y ( " M a t h S q r " ) : f l g R e p l a c e = 0
000572 WHEN &h23: U n a r y ( " M a t h I n v " ) : f l g R e p l a c e = 0
000573 WHEN &h24: i z e : S t a c k R o t : f l g R e p l a c e = 0
000574 WHEN &h25: K S U B
000575 WHEN &h26: F i n a l i z e : R e a d P R G M : f l g R e p l a c e = 0
000576 WHEN &h27: K D I V
000577 WHEN &h28: i z e : D o A b s : f l g R e p l a c e = 0
000578 WHEN &h29: i z e : D o S g n : f l g R e p l a c e = 0
000580 WHEN &h2a: i z e : D o I n t : f l g R e p l a c e = 0
000581 WHEN &h2b: i z e : D o F r a c : f l g R e p l a c e = 0
000582 WHEN &h2c: i z e : D o M a x : f l g R e p l a c e = 0
000583 WHEN &h2d: i z e : D o A n d : f l g R e p l a c e = 0
000584 WHEN &h2e: i z e : D o O r : f l g R e p l a c e = 0
000585 WHEN &h2f: i z e : D o X o r : f l g R e p l a c e = 0
000586 WHEN &h30: i z e : D o N o t : f l g R e p l a c e = 0
000587 WHEN &h31: i z e : D o R n d : f l g R e p l a c e = 0
000588 WHEN &h32: i z e : f l g R e p l a c e = 0
000589 WHEN &h33: K S C R
000590 ENDCASE
000591 ENDIF
000592 END
000593

```