```
/*

     001-Generated C code for functional specification 'RECEIVE_MSGS'

       VERSION: 3.2.3.9 C-RAT
              : 1.6.0.0
        AUTHOR: Hamilton Technologies Inc. Copyright 1991-2015.
     OPERATION: RECEIVE_MSGS
     GENERATED: Sat Sep 12 16:02:09 2015

        SCCSID: @(#) %M% %I% of %G%.

*/
#include "MSGS.h"
#include "OMAP.h"
#include "STR.h"
#include "BOOLEAN.h"
#include "CPORT.h"
#include "NAT.h"
#include "CHAR.h"
#include "TMAP.h"

#include <stdio.h>
#include <math.h>
#include <errno.h>
#include "BOOLEAN.h"
#include "NAT.h"

fRECEIVE_MSGS(V0CP0,V0WAIT,
              V0MS)
IDECLARE_CPORT(V0CP0)
IDECLARE_NAT(V0WAIT)
ODECLARE_MSGS(V0MS)
{
                /* __LOCAL_VARIABLE_DECLARATIONS__ */
DECLARE_CPORT(V0CP1)
```

```
DECLARE_STR(V0OMSTR1)
DECLARE_BOOLEAN(V0N)
DECLARE_BOOLEAN(V0E)
DECLARE_BOOLEAN(V0R)
DECLARE_BOOLEAN(D0D1)
DECLARE_BOOLEAN(D0D4)
DECLARE_BOOLEAN(D0D8)
DECLARE_MSGS(V1_SET2)
DECLARE_STR(C13)
DECLARE_CHAR(C14)
DECLARE_OMAP(V1_OM)
DECLARE_STR(V1_ONE_OMAP_STR)
DECLARE_CHAR(C17)
DECLARE_STR(V1_OMSTR2)
DECLARE_CHAR(C19)
DECLARE_STR(V1_OMSTR20)
DECLARE_NAT(V1_Z)
DECLARE_NAT(V1_NUM)
DECLARE_STR(V1_OMSTR1)
DECLARE_STR(V1_NUMSTR)
DECLARE_CHAR(C25)
DECLARE_STR(V1_OMSTR00)
DECLARE_CHAR(C27)
DECLARE_CHAR(C28)
DECLARE_STR(V1_OMSTRM)
DECLARE_OMAP(V1_OMSM)
DECLARE_STR(V1_OMSTR)
DECLARE_OMAP(V1_OMS)
DECLARE_STR(V1_OMSTR01)
DECLARE_CHAR(C34)
DECLARE_OMAP(V1_OMS0)
DECLARE_MSGS(V1_SET1)
DECLARE_TMAP(V1_TM)
DECLARE_BOOLEAN(D1_D12)
DECLARE_BOOLEAN(V2_0ISRJ)
DECLARE_BOOLEAN(V2_0ISE)
```

```
DECLARE_BOOLEAN(D2_0D4)
DECLARE_BOOLEAN(D2_0D5)
                /* __ITERATION_VARIABLE_DECLARATIONS__ */
int rec18REPEAT;
DECLARE_OMAP(R181_OMS0)
DECLARE_STR(R181_OMSTR01)
DECLARE_TMAP(R181_TM)
                /* __CONSTANT_DECLARATIONS_AND_ASSIGNMENTS__ */
NEWSTACK_TRACE_IDSC("RECEIVE_MSGS");
DOT_K_STR("\0",C13)
DOT_K_CHAR('>',C14)
DOT_K_CHAR('L',C17)
DOT_K_CHAR('R',C19)
DOT_K_CHAR('L',C25)
DOT_K_CHAR('#',C27)
DOT_K_CHAR('<',C28)
DOT_K_CHAR('<',C34)
                /* __FUNCTION_SOURCE_CODE_BEGINNING__ */
   ISREJECT_CPORT(V0CP0,D0D1)
     if(D0D1<1)
 {if(D0D1 == REJECT_BOOLEAN) {REJECT_TEST_BOOLEAN()}
   fREAD_ERROR_SELECT(V0CP0,V0WAIT,&V0R,&V0E,&V0N);
   CLONE_BOOLEAN(V0R,D0D4)
     if(D0D4<1)
 {if(D0D4 == REJECT_BOOLEAN) {REJECT_TEST_BOOLEAN()}
   K_MSGS(V0CP0,*V0MS)
}/*FALSE*/
else{/*CAN_RECEIVE*/
   fRECEIVE_A_STR_CPORT(V0CP0,&V0OMSTR1,&V0CP1);
   ISREJECT_CPORT(V0CP1,D0D8)
     if(D0D8<1)
 {if(D0D8 == REJECT_BOOLEAN) {REJECT_TEST_BOOLEAN()}
   READTMAP_MSGS(V0OMSTR1,V1_TM)
   K_MSGS(V1_TM,V1_SET1)
   CONVERT_OMAP_MSGS(V1_SET1,V1_OMS0)
   NEXT_STR(C34,V0OMSTR1,V1_OMSTR01)
```

```
   IS_EMPTY_STR(V1_OMSTR01,V2_0ISE)
   IS_REJECT_STR(V1_OMSTR01,V2_0ISRJ)
   OR_BOOLEAN(V2_0ISRJ,V2_0ISE,D2_0D4)
     if(D2_0D4<1)
 {if(D2_0D4 == REJECT_BOOLEAN) {REJECT_TEST_BOOLEAN()}
R181_OMS0=V1_OMS0;
R181_OMSTR01=V1_OMSTR01;
R181_TM=V1_TM;
rec18REPEAT=1;
while(rec18REPEAT--){
   ATNULL_STR(V1_OMSTR01,D2_0D5)
     if(D2_0D5<1)
 {if(D2_0D5 == REJECT_BOOLEAN) {REJECT_TEST_BOOLEAN()}
   CHARMATCH_STR(C28,C27,V1_OMSTR01,V1_OMSTR00)
   SPLIT_STR(C25,V1_OMSTR00,V1_NUMSTR,V1_OMSTR1)
   CONVERT_NAT_STR(V1_NUMSTR,V1_NUM)
   SIZE_STR(V1_OMSTR1,V1_Z)
   LT_NAT(V1_Z,V1_NUM,D1_D12)
     if(D1_D12<1)
 {if(D1_D12 == REJECT_BOOLEAN) {REJECT_TEST_BOOLEAN()}
   LOCATE_STR(V1_NUM,V1_OMSTR1,V1_OMSTR20)
   NEXT_STR(C19,V1_OMSTR20,V1_OMSTR2)
   SPLIT_STR(C17,V1_OMSTR2,V1_ONE_OMAP_STR,V1_OMSTRM)
   CONVERT_OMAP_STR(V1_ONE_OMAP_STR,V1_TM,V1_OM)
   MERGE_OMAP(C14,V1_OM,V1_OMS0,V1_OMSM)
}/*FALSE*/
else{/*NOT_ENOUGH_ER*/
   K_STR(C13,V1_OMS0,V1_OMSTRM)
   CLONE_OMAP(V1_OMS0,V1_OMSM)
}/*TRUE*/
     rec18REPEAT=1;
     V1_OMS0=V1_OMSM;
     V1_OMSTR01=V1_OMSTRM;
     V1_TM=V1_TM;
}/*FALSE*/
else{/*ATNULL*/
```

```
   CLONE_STR(V1_OMSTR01,V1_OMSTR)
   CLONE_OMAP(V1_OMS0,V1_OMS)
}/*TRUE*/
}
V1_OMS0=R181_OMS0;
V1_OMSTR01=R181_OMSTR01;
V1_TM=R181_TM;
}/*FALSE*/
else{/*ISEMPTY*/
   CLONE_STR(V1_OMSTR01,V1_OMSTR)
   CLONE_OMAP(V1_OMS0,V1_OMS)
}/*TRUE*/
   CONVERT_MSGS_OMAP(V1_OMS,V1_SET2)
   CLONE_MSGS(V1_SET2,*V0MS)
}/*FALSE*/
else{/*K_MSGS*/
   K_MSGS(V0OMSTR1,*V0MS)
}/*TRUE*/
}/*TRUE*/
}/*FALSE*/
else{/*K_MSGS*/
   K_MSGS(V0CP0,*V0MS)
}/*TRUE*/
ENDSTACK_IDSC(); /* STR Garbage Collector */

return;
}
/*  -------------- end of source ------------------*/
```