AmigaMail

COLLABORATORS

	TITLE : AmigaMail	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 23, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

AmigaMail

1.1 Developer Support Package / copybuff.doc

```
TABLE OF CONTENTS

   any_sana2_protocol/CopyFromBuff
   any_sana2_protocol/CopyToBuff
```

1.2 any_sana2_protocol/CopyFromBuff

```
NAME
    CopyFromBuff -- Copy n bytes from an abstract data structure.

SYNOPSIS
    success = CopyFromBuff(to, from, n)
    d0                     a0  a1    d0

    BOOL CopyToBuff(VOID *, VOID *, ULONG);

FUNCTION
    This function copies 'n' bytes of data in the abstract data structure
    pointed to by 'from' into the contigous memory pointed to by 'to'.
    'to' must contain at least 'n' bytes of usable memory or innocent
    memory will be overwritten.

INPUTS
    to              - pointer to contiguous memory to copy to.
    from            - pointer to abstract structure to copy from.
    n               - number of bytes to copy.

RESULT
    success         - TRUE if operation was successful, else FALSE.

EXAMPLE

NOTES
    This function must be callable from interupts.  In particular, this
    means that this function may not directly or indirectly call any
```

```
    system memory functions (since those functions rely on Forbid() to
    protect themselves) and that  you must not compile this function
    with stack checking enabled.  See the RKM:Libraries Exec:Interupts
    chapter for more details on what is legal in a routine called from
    an interupt handler.

    'C' programmers should not compile with stack checking (option '-v'
    in SAS) and should geta4() or __saveds.

BUGS

SEE ALSO
```

1.3 any_sana2_protocol/CopyToBuff

```
NAME
    CopyToBuff -- Copy n bytes to an abstract data structure.

SYNOPSIS
    success = CopyToBuff(to, from, n)
    d0                   a0  a1    d0

    BOOL CopyToBuff(VOID *, VOID *, ULONG);

FUNCTION
    This function first does any initialization and/or allocation
    required to prepare the abstract data structure pointed at by 'to'
    to be filled with 'n' bytes of data from 'from'.  It then executes
    the copy operation.

    If, for example, there is not enough memory available to prepare
    the abstract data structure, the call is failed and FALSE is returned.

    The buffer management scheme should be such that any memory needed
    to fulfill CopyToBuff() calls is already allocated from the system
    before the call to CopyToBuff() is made.

INPUTS
    to              - pointer to abstract structure to copy to.
    from            - pointer to contiguous memory to copy from.
    n               - number of bytes to copy.

RESULT
    success         - TRUE if operation was successful, else FALSE.

EXAMPLE

NOTES
    This function must be callable from interupts.  In particular, this
    means that this function may not directly or indirectly call any
    system memory functions (since those functions rely on Forbid() to
    protect themselves) and that you must not compile this function
    with stack checking enabled.  See the RKM:Libraries Exec:Interupts
    chapter for more details on what is legal in a routine called from
    an interupt handler.
```

```
    'C' programmers should not compile with stack checking (option '-v'
    in SAS) and should geta4() or __saveds.

BUGS

SEE ALSO
```