Message Mailbox Management

中興大學資訊科學系
指導教授：張軒彬
學生：陳奇進

Message Mailbox Management

- Introduction
- µC/OS-II provides seven services
  - OSMboxCreate()
  - OSMboxDel()
  - OSMboxPend()
  - OSMboxPost()
  - OSMboxPostOpt()
  - OSMboxAccept()
  - OSMboxQuery()

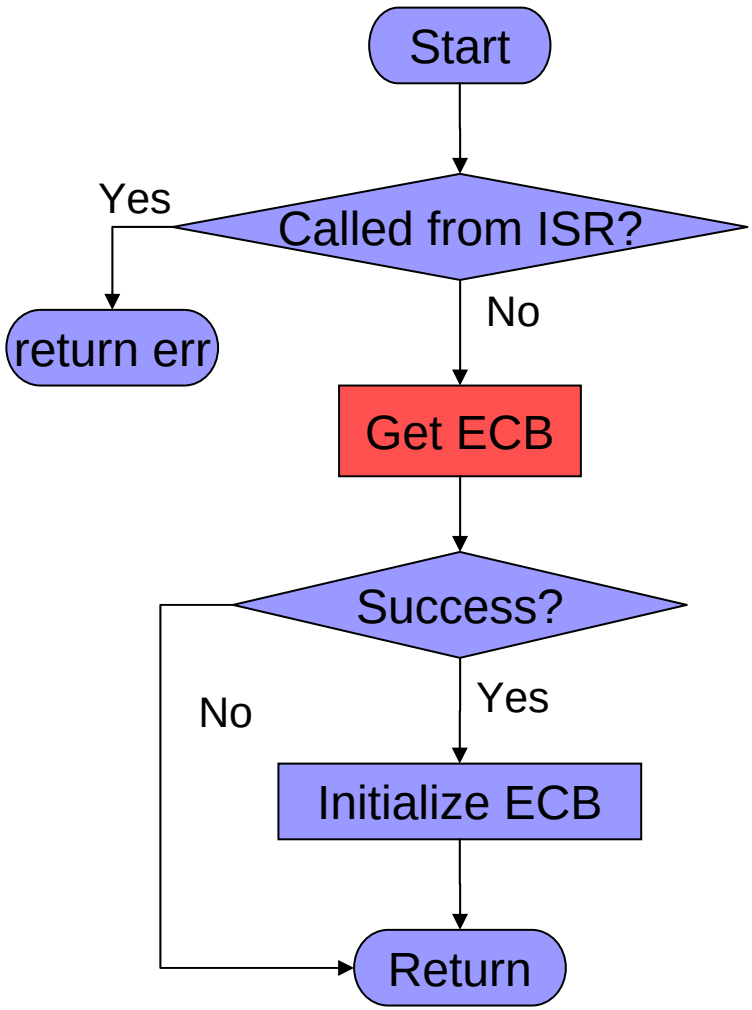
Introduction

- A message mailbox is a µC/OS-II object that allows a task or an ISR to send a pointer-sized variable to another task.
- The msg pointer would typically be initialized to point to some application specific data structure containing a message
- A mailbox can only contain one pointer (mailbox is full) or a pointer to **NULL** (mailbox is empty).

OSMboxCreate()

- **Prototype：** `OS_EVENT *OSMboxCreate(void *msg)`
- **Description：** This function creates a message mailbox if free ECB are available.
- **Arguments：**
  - `msg` is a pointer to a message that you wish to deposit in the mailbox.
- **Returns：**
  - `!= (OS_EVENT *)0` is a pointer to the event control clock
  - `== (OS_EVENT *)0` if no event control blocks were available

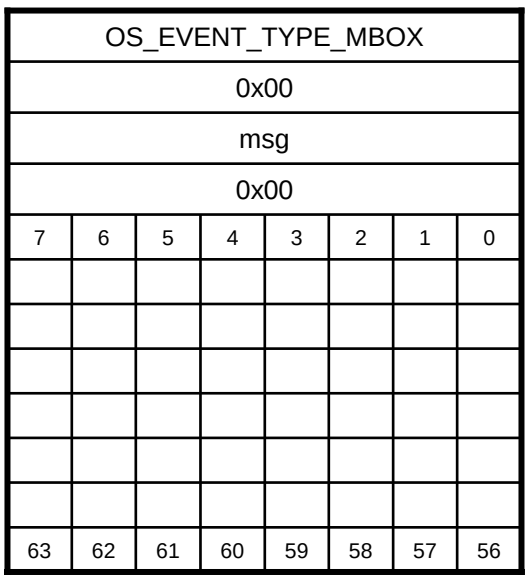
OSMboxCreate()




```
OS_EVENT  *pevent;

if (OSIntNesting > 0)
{   return ((OS_EVENT *)0);          }

pevent = OSEventFreeList;
if (OSEventFreeList != (OS_EVENT *)0)
{   OSEventFreeList = (OS_EVENT *)
    OSEventFreeList->OSEventPtr;     }

if (pevent != (OS_EVENT *)0)
{   pevent->OSEventType= OS_EVENT_TYPE_MBOX;
    pevent->OSEventCnt=0;
    pevent->OSEventPtr= msg;
    OS_EventWaitListInit(pevent);    }
return (pevent);
```

OSMboxCreate()

OS_EVENT



OSEventType
OSEventCnt
OSEventPtr
OSEventGrp
OSEventTbl[]

All
Initialized
To
0x00

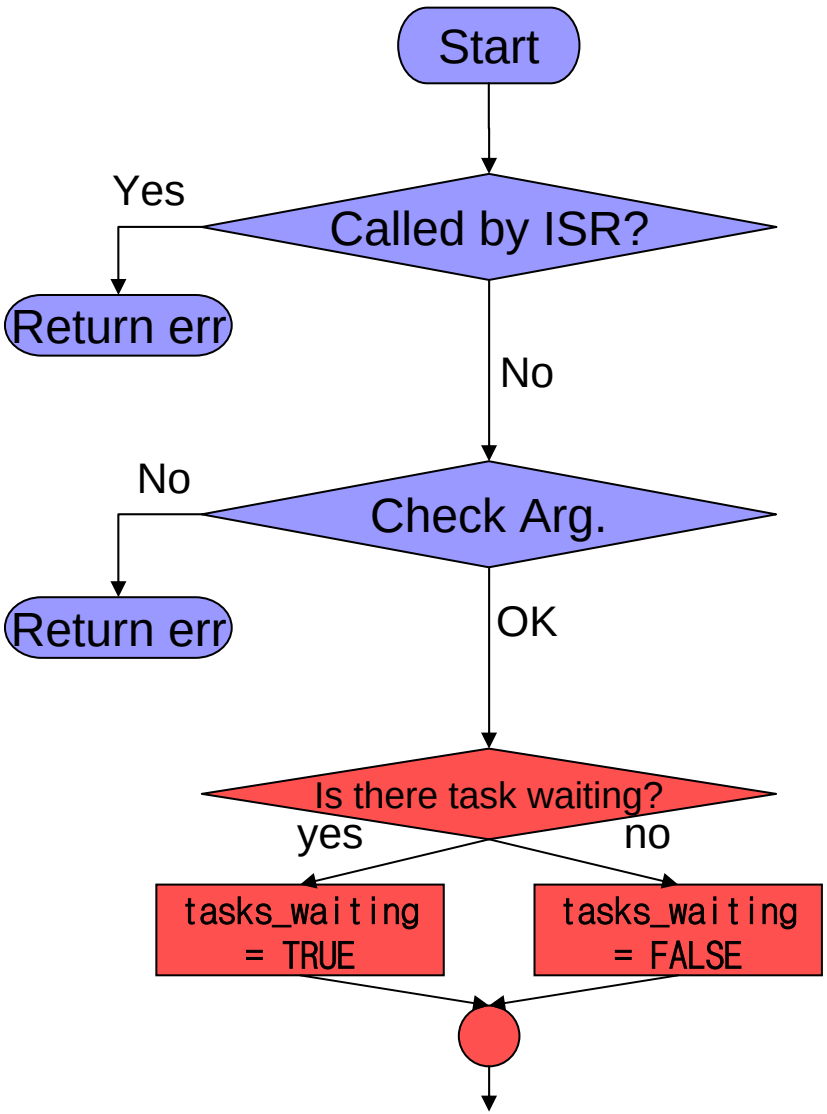
OSMboxDel()

- **Prototype**：`OS_EVENT *OSMboxDel(OS_EVENT *pevent, INT8U opt, INT8U *err)`
- **Description**： This function deletes a mailbox and readies all tasks pending on the mailbox.
- **Arguments**：
  - pevent is a pointer to the ECB.
  - opt determines delete options as follows
    - `OS_DEL_NO_PEND` Delete the mailbox ONLY if no task pending.
    - `OS_DEL_ALWAYS` Deletes the mailbox even if tasks are waiting.

OSMboxDel()

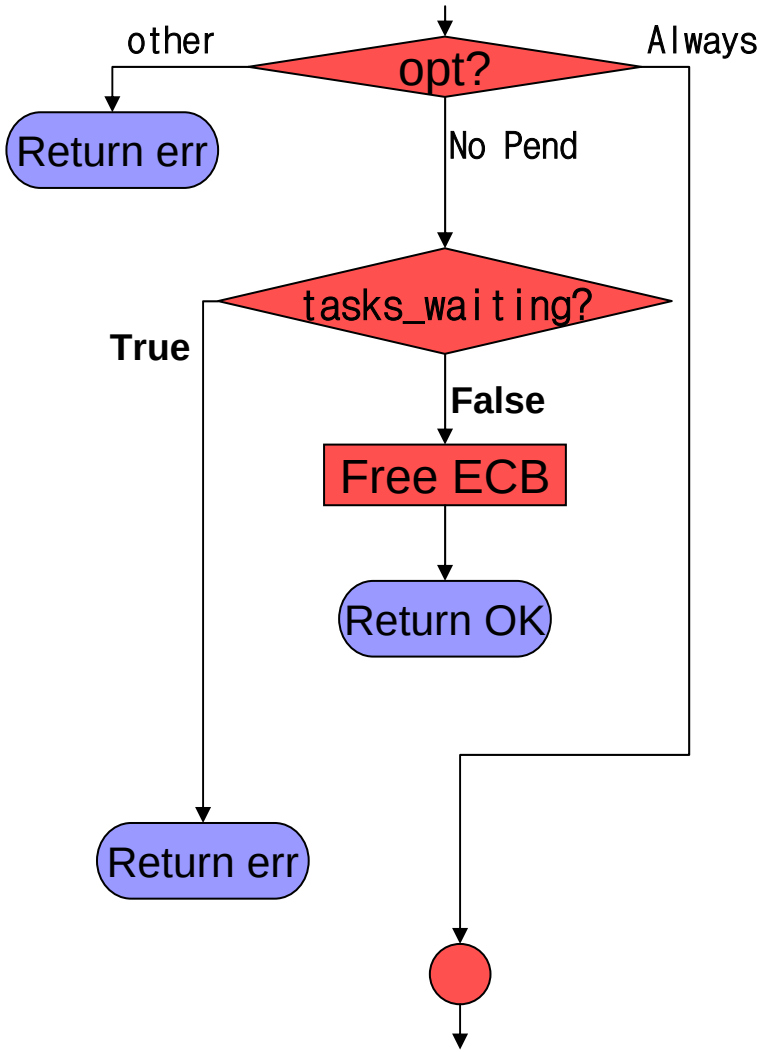
- **Arguments：**
  - err　is a pointer to an error code that can contain one of the following values:
    - OS_NO_ERR
    - OS_ERR_DEL_ISR
    - OS_ERR_INVALID_OPT
    - OS_ERR_TASK_WAITING
    - OS_ERR_EVENT_TYPE
    - OS_ERR_PEVENT_NULL
- **Returns：**
  - peven　upon error
  - (OS_EVENT *)0　if the mailbox was successfully deleted.

OSMboxDel()



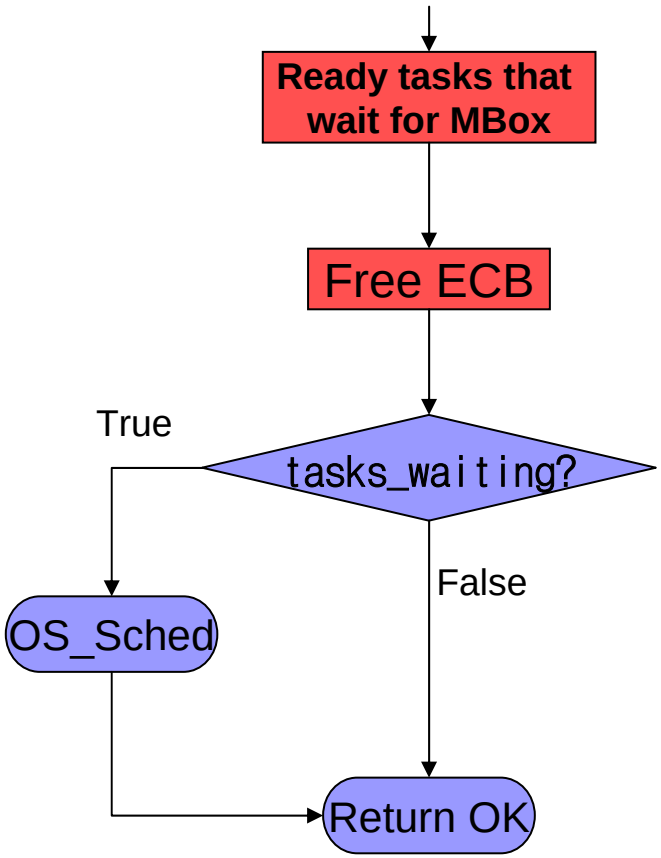

```
BOOLEAN    tasks_waiting;

if (OSIntNesting > 0)
{   *err = OS_ERR_DEL_ISR;
    return (pevent);        }

if (pevent == (OS_EVENT *)0)
{   *err = OS_ERR_PEVENT_NULL;
    return (pevent);                    }
if (pevent->OSEventType != S_EVENT_TYPE_MBOX)
{   *err = OS_ERR_EVENT_TYPE;
    return (pevent);                    }

if (pevent->OSEventGrp != 0x00)
{   tasks_waiting = TRUE;   }
else
{   tasks_waiting = FALSE;  }
```

OSMboxDel()




```
switch (opt)
{
  case OS_DEL_NO_PEND:
    if (tasks_waiting == FALSE)
    {
          pevent->OSEventType =
                    OS_EVENT_TYPE_UNUSED;
          pevent->OSEventPtr  = OSEventFreeList;
          OSEventFreeList     = pevent;
          *err = OS_NO_ERR;
          return ((OS_EVENT *)0);
    }
    else
    {
          *err = OS_ERR_TASK_WAITING;a
          return (pevent);
    }
```

OSMboxDel()




```
case OS_DEL_ALWAYS:
    while (pevent->OSEventGrp != 0x00)
        OS_EventTaskRdy(pevent, (void *)0,
                        OS_STAT_MBOX);
    pevent->OSEventType = OS_EVENT_TYPE_UNUSED;
    pevent->OSEventPtr  = OSEventFreeList;
    OSEventFreeList     = pevent;
    if (tasks_waiting == TRUE)
        OS_Sched();
    *err = OS_NO_ERR;
    return ((OS_EVENT *)0);

default:
    *err = OS_ERR_INVALID_OPT;
    return (pevent);
}
```

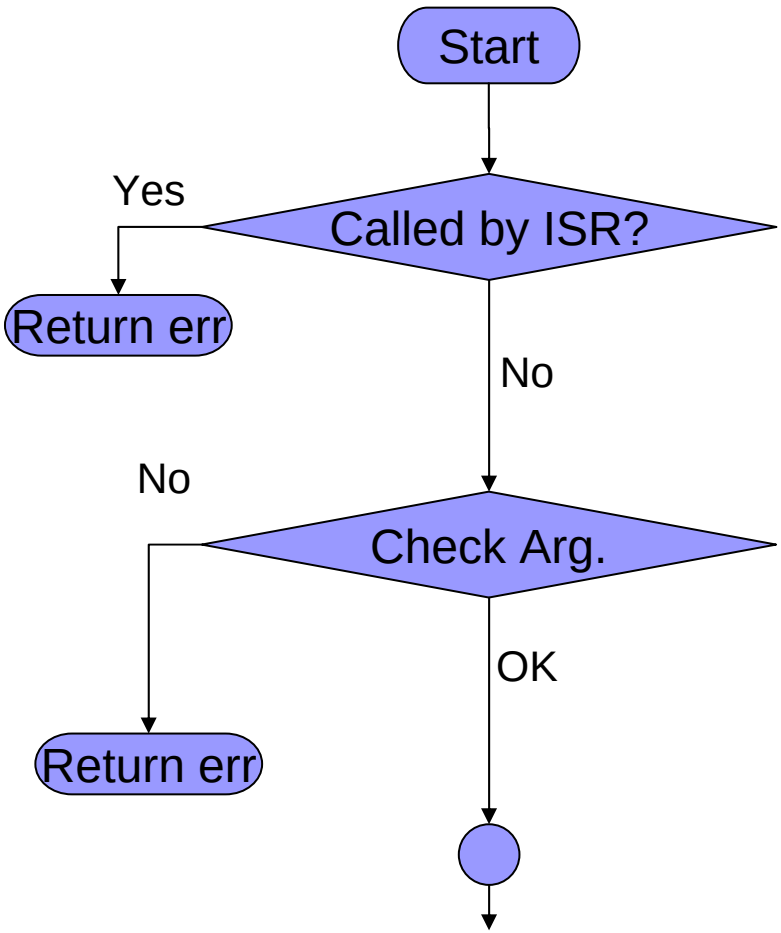
OSMboxPend()

- **Prototype**：`void *OSMboxPend(OS_EVENT *pevent, INT16U timeout, INT8U *err)`
- **Description**： This function waits for a message to be sent to a mailbox.
- **Arguments**：
  - `pevent` is a pointer to the ECB.
  - `timeout` is an optional timeout period (in clock ticks).
  - `err` is a pointer to where an error message will be deposited.
    - `OS_NO_ERR`
    - `OS_TIMEOUT`
    - `OS_ERR_EVENT_TYPE`
    - `OS_ERR_PEND_ISR`
    - `OS_ERR_PEVENT_NULL`

OSMboxPend()

- **Returns**：
  - `!= (void *)0` is a pointer to the message received
  - `== (void *)0`
    - if no message was received
    - if 'pevent' is a NULL pointer
    - if you didn't pass the proper pointer to the ECB.

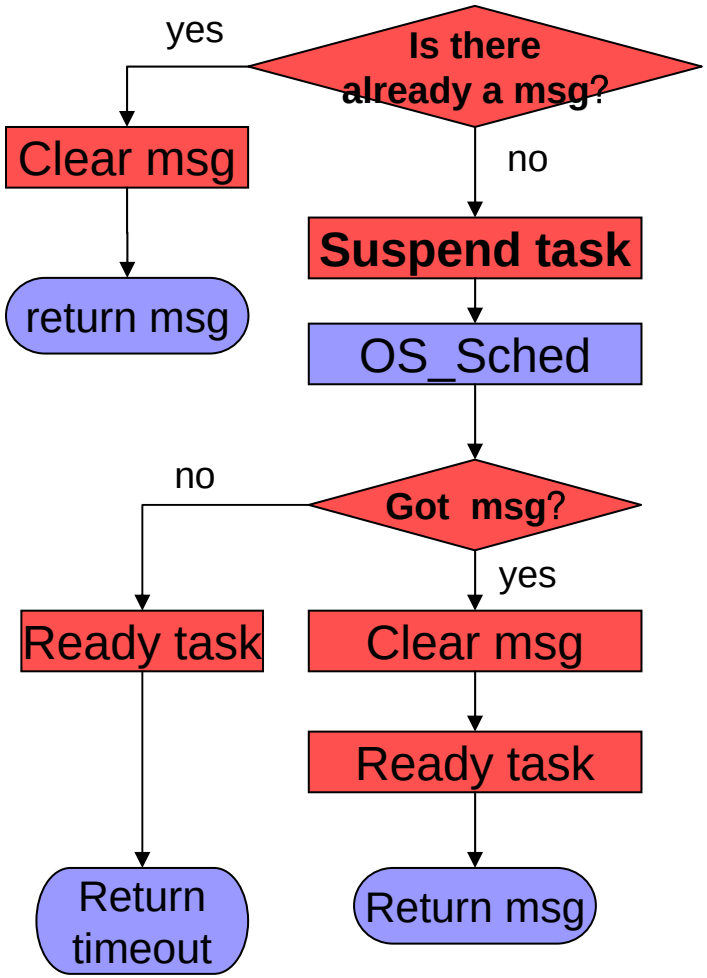
OSMboxPend()




```
if (OSIntNesting > 0)
{  *err = OS_ERR_DEL_ISR;
   return (pevent);     }
```

```
if (pevent == (OS_EVENT *)0)
{  *err = OS_ERR_PEVENT_NULL;
   return (pevent);                }
if (pevent->OSEventType !=
   S_EVENT_TYPE_MBOX)
{  *err = OS_ERR_EVENT_TYPE;
   return (pevent);                }
```

OSMboxPend()

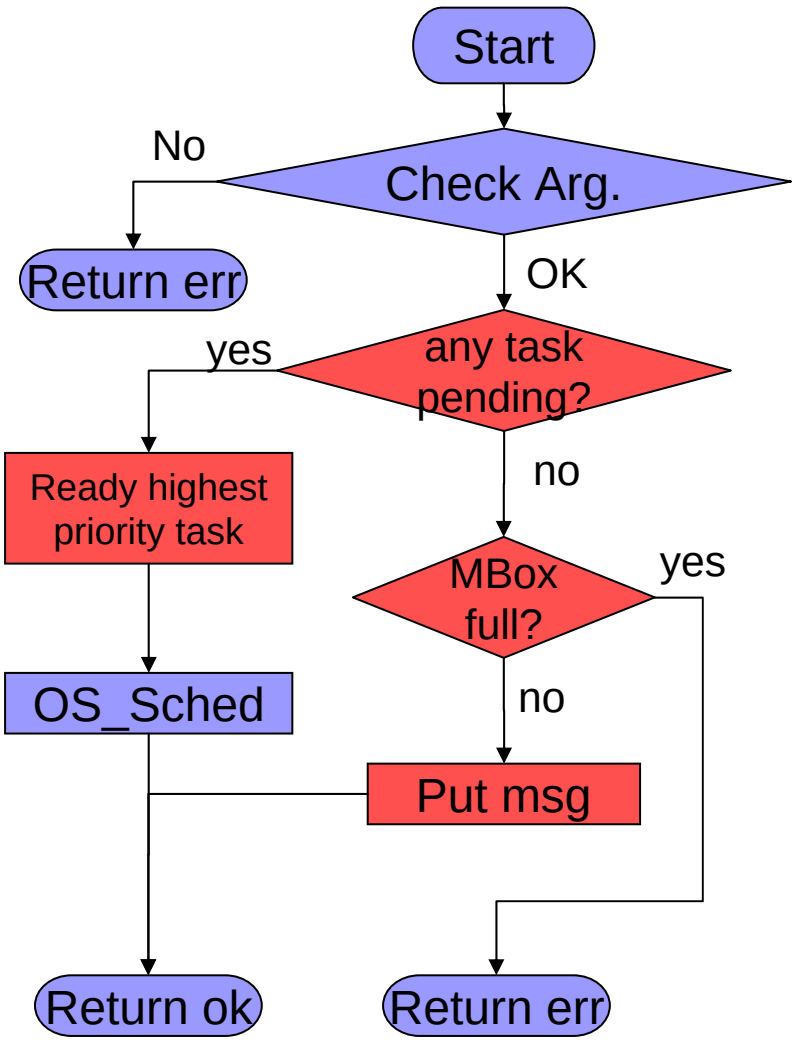



```
msg = pevent->OSEventPtr;
if (msg != (void *)0)
{
    pevent->OSEventPtr = (void *)0;
    *err = OS_NO_ERR;
    return (msg);
}
OSTCBCur->OSTCBStat |= OS_STAT_MBOX;
OSTCBCur->OSTCBDly   = timeout;
OS_EventTaskWait(pevent);
OS_Sched();
msg = OSTCBCur->OSTCBMsg;
if (msg != (void *)0)
{
    OSTCBCur->OSTCBMsg      = (void *)0;
    OSTCBCur->OSTCBStat     = OS_STAT_RDY;
    OSTCBCur->OSTCBEventPtr = (OS_EVENT *)0;
    *err                    = OS_NO_ERR;
    return (msg);
}
OS_EventTO(pevent);
*err = OS_TIMEOUT;
return ((void *)0);
```

OSMboxPost ()

- **Prototype**： `INT8U OSMboxPost (OS_EVENT *pevent, void *msg)`
- **Description**：This function sends a message to a mailbox.
- **Arguments**：
  - `pevent` is a pointer to the event control block associated with the desired mailbox.
  - `msg` is a pointer to the message to send.
- **Returns**：
  - `OS_NO_ERR`
  - `OS_MBOX_FULL`
  - `OS_ERR_EVENT_TYPE`
  - `OS_ERR_PEVENT_NULL`
  - `OS_ERR_POST_NULL_PTR`

OSMboxPost()

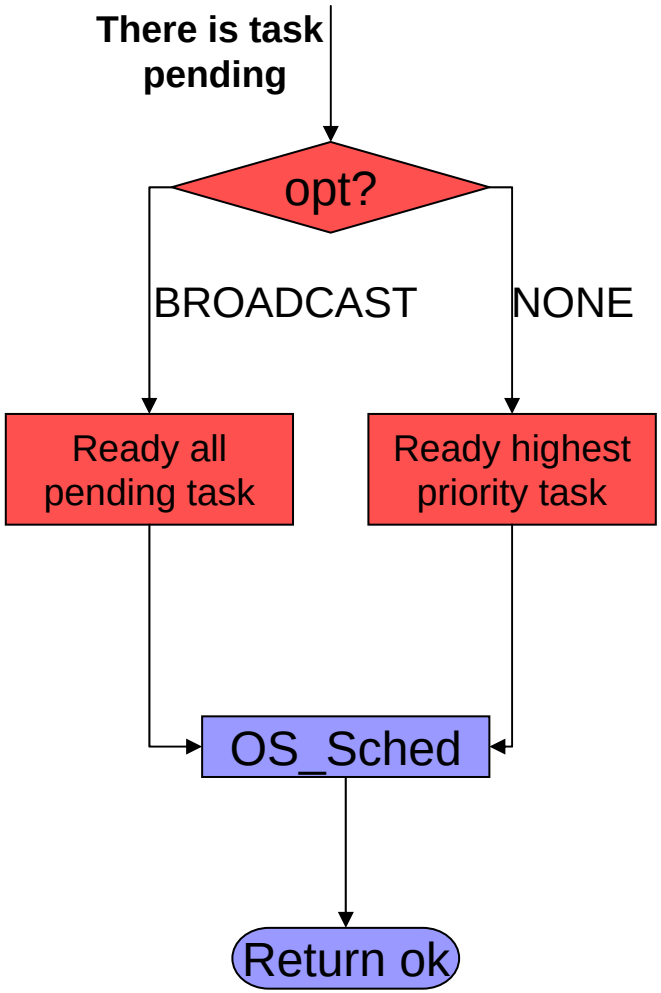



```
if (pevent->OSEventGrp != 0x00)
{
    OS_EventTaskRdy(pevent,msg,OS_STAT_MBOX);
    OS_Sched();
    return (OS_NO_ERR);
}

if (pevent->OSEventPtr != (void *)0)
    return (OS_MBOX_FULL);

pevent->OSEventPtr = msg;
return (OS_NO_ERR);
```

OSMboxPostOpt ()

- **Prototype：** `INT8U OSMboxPostOpt(OS_EVENT *pevent, void *msg, INT8U opt)`
- **Description：** Extended OSMboxPost()
- **Arguments：**
  - `pevent` is a pointer to the ECB associated with the desired mailbox.
  - `msg` is a pointer to the message to send.
  - `opt` determines the type of POST performed:
    - `OS_POST_OPT_NONE` POST to a single waiting task
    - `OS_POST_OPT_BROADCAST` POST to ALL tasks that are waiting on the mailbox
- **Returns：** the same as OSMboxPost()

OSMboxPostOpt()

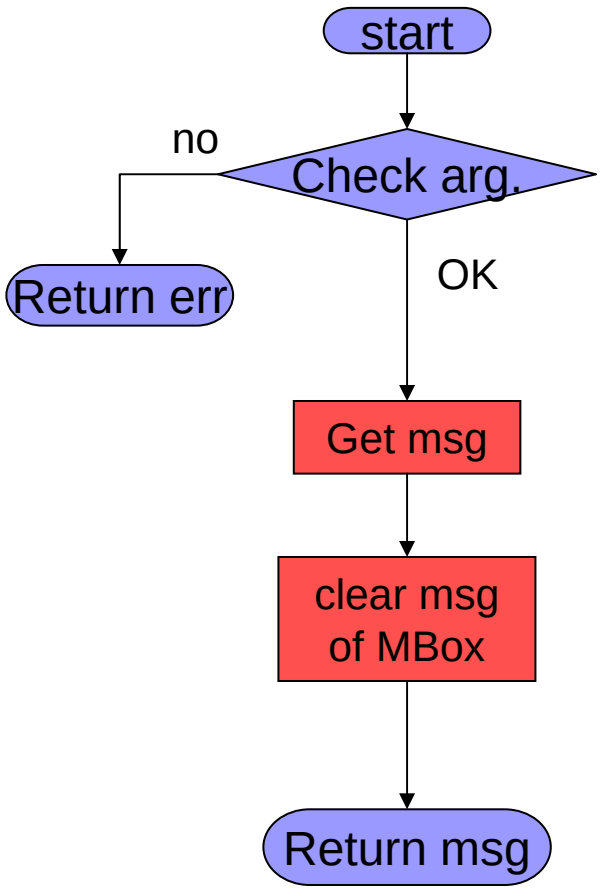



```
if ((opt & OS_POST_OPT_BROADCAST) != 0x00)
{
    while (pevent->OSEventGrp != 0x00)
        OS_EventTaskRdy(pevent, msg, OS_STAT_MBOX);
}
else
{
  OS_EventTaskRdy(pevent, msg, OS_STAT_MBOX);
}

OS_Sched();
return (OS_NO_ERR);
```

OSMboxAccept ()

- **Prototype**：`void *OSMboxAccept (OS_EVENT *pevent)`
- **Description**：This function checks the mailbox to see if a message is available.
- **Arguments**：
  - `pevent` is a pointer to the event control block.
- **Returns**：
  - `!= (void *)0` is the message in the mailbox if one is available.
  - `== (void *)0`
    - if the mailbox is empty
    - if 'pevent' is a NULL pointer
    - if you didn't pass the proper pointer to the ECB.

OSMboxAccept()




```
void      *msg;

#if OS_ARG_CHK_EN > 0
    if (pevent == (OS_EVENT *)0)
        return ((void *)0);
    if (pevent->OSEventType != OS_EVENT_TYPE_MBOX)
        return ((void *)0);
#endif

    msg                = pevent->OSEventPtr;
    pevent->OSEventPtr = (void *)0;
    return (msg);
```

OSMboxQuery()

- **Prototype**：`INT8U OSMboxQuery (OS_EVENT *pevent, OS_MBOX_DATA *pdata)`
- **Description**：This function obtains information about a message mailbox.
- **Arguments**：
  - `pevent` is a pointer to the event control block.
  - `pdata` is a pointer to a structure that will contain information about the message mailbox.
- **Returns**：
  - `OS_NO_ERR`
  - `OS_ERR_EVENT_TYPE`
  - `OS_ERR_PEVENT_NULL`

OSMboxQuery()

```
typedef struct
{
    void   *OSMsg;
    INT8U   OSEventTbl[OS_EVENT_TBL_SIZE];
    INT8U   OSEventGrp;
} OS_MBOX_DATA;
```

OSMboxQuery()

```
INT8U                   *psrc;
NT8U                    *pdest;
pdata->OSEventGrp       = pevent->OSEventGrp;
psrc                    = &pevent->OSEventTbl[0];
pdest                   = &pdata->OSEventTbl[0];


#if OS_EVENT_TBL_SIZE > 0
    *pdest++            = *psrc++;
#endif
           :
           :
#if OS_EVENT_TBL_SIZE > 7
    *pdest              = *psrc;
#endif

pdata->OSMsg = pevent->OSEventPtr;
return (OS_NO_ERR);
```

Using a Mailbox as a Binary Semphore

- Initializing mailbox with a pointer (void *)1
- Signal：OSMboxPost()
- Wait：OSMboxPend()

Using a Mailbox Instead of OSTimeDly()

- Example：
  - OSMboxPend(MboxTimeDly, TIMEOUT, &err);