Object Objects

Introduction

Each **Object** shares some common functionality with other objects. The Object module encapsulates the common object methods and macros into a single module. This makes object handling consistent and more efficient.

Methods

Create, Delete, Validate Methods

mpiObject**Validate**

Configuration and Information Methods

mpiObject**ModuleId**

mpiObject**TimeoutGet**

mpiObject**TimeoutSet**

meiObject**TraceGet**

meiObject**TraceSet**

Data Types

MPIObject**Map**

Macros

mpiObjectMap**AND_ASSIGN**

mpiObjectMap**ASSIGN**

mpiObjectMap**BitCountMAX**

mpiObjectMap**BitGET**

mpiObjectMap**BitSET**

mpiObjectMap**CLEAR**

mpiObjectMap**COMPLEMENT**

mpiObjectMap**IS_CLEAR**

mpiObjectValidate

Declaration

```
long mpiObjectValidate(MPIHandle   handle,
                       MPIModuleId moduleId)
```

Required Header

stdmpi.h

Description

The **ObjectValidate** method verifies that the object (handle) is of the type moduleId.

Return Values	
MPIMessageOK	if ***handle*** is a handle to a valid object of type ***moduleId***.

See Also

mpiObjectModuleId

Declaration

```
long mpiObjectModuleId(MPIHandle    handle,
                       MPIModuleId  *moduleId)
```

Required Header

stdmpi.h

Description

The **ObjectModuleId** function writes the MPIModuleId of the object (***handle***) to the location pointed to by ***moduleId***.

Return Values

MPIMessageOK	if *ObjectModuleId* successfully writes the MPIModuleId to the location

See Also

mpiObjectTimeoutGet

Declaration

```
long mpiObjectTimeoutGet(MPIHandle handle,
                         MPIWait   *timeout)
```

Required Header stdmpi.h

Description The **ObjectTimeoutGet** method gets the timeout value for the object (***handle***), and writes it to the location pointed to by ***timeout***.

Return Values	
MPIMessageOK	if *ObjectTimeoutGet* successfully writes the timeout value to the location

See Also mpiObjectTimeoutSet

mpiObjectTimeoutSet

Declaration

```
long mpiObjectTimeoutSet(MPIHandle handle,
                         MPIWait   timeout)
```

Required Header

stdmpi.h

Description

ObjectTimeoutSet sets the timeout value for an object (***handle***) to ***timeout***.

The ***timeout*** value is used in a multi-threaded environment when an MPI function must block to wait for a shared resource to become available. The default ***timeout*** value is MPIWaitFOREVER.

If "timeout" is	*Then*
MPIWaitFOREVER	*ObjectTimeoutSet* will wait until the resource becomes available
MPIWaitPOLL	*ObjectTimeoutSet* will not wait for the resource to become available. If the shared resource is not available, a timeout will be considered to have occurred.
a value	*ObjectTimeoutSet* will wait timeout milliseconds for the resource to become available

Return Values

MPIMessageOK	if ObjectTimeoutSet successfully sets the timeout value of the object
MPIMessageTIMEOUT	if a timeout occurs before the shared resource becomes available

See Also

mpiObjectTimeoutGet

meiObjectTraceGet / meiObjectTraceGET

meiObjectTraceGet

Declaration

```
long meiObjectTraceGet(MPIHandle     handle,
                       MEITraceMask  *traceMask)
```

Required Header stdmei.h

Description **ObjectTraceGet** gets an Object's trace mask and writes it to the value pointed to by traceMask.

handle	a handle to an object
***traceMask**	a pointer to the value of an object's trace mask

Return Values	
MPIMessageOK	if *ObjectTraceGet* successfully gets the trace mask for an object.

meiObjectTraceGET

Declaration

```
#define  meiObjectTraceGET(object)      (((MEIObject)(object))->trace)
```

Required Header stdmei.h

Description **ObjectTraceGET** gets the object's global trace mask.

See Also meiObjectTraceSET | meiObjectTraceSet

meiObjectTraceSet / meiObjectTraceSET

meiObjectTraceSet

Declaration

```
long meiObjectTraceSet(MPIHandle     handle,
                       MEITraceMask  traceMask)
```

Required Header

stdmei.h

Description

ObjectTraceSet sets an Object's trace mask to the value specified by ***traceMask***.

handle	a handle to an object
traceMask	the value of an object's trace mask

Return Values

MPIMessageOK	if *ObjectTraceSet* successfully sets the trace mask for an object.

meiObjectTraceSET

Declaration

```
#define  meiObjectTraceSET(object,mask)
                          (((MEIObject)(object))->trace = (mask))
```

Required Header

stdmei.h

Description

ObjectTraceSET sets the object's global trace mask.

See Also

meiObjectTraceGET | meiObjectTraceGet

MPIObjectMap

MPIObjectMap

```
typedef unsigned long   MPIObjectMap;
```

Description

MPIObjectMap	A map of MPI objects. An ObjectMap is a bitmap, where each numbered bit represents the presence or absence of the correspondingly numbered object. Valid maps are Axis/Filter and Filter/Motor. The Axis/Filter map can also be read, but setting this map must be done through the corresponding Filter objects.

See Also

mpiObjectMapAND_ASSIGN

Declaration

```
#define   mpiObjectMapAND_ASSIGN(dst,src)       ((dst) &= (src))
```

Required Header

stdmpi.h

Description

Bitwise ANDs the dst object map with the ***src*** object map and assigns the result to ***dst***.

See Also

mpiObjectMapASSIGN

mpiObjectMapASSIGN

Declaration

```
#define    mpiObjectMapASSIGN(dst,src)   ((dst) = (src))
```

Required Header

stdmpi.h

Description

ObjectMapASSIGN assigns ***src*** object map to the ***dst*** object map.

See Also

mpiObjectMapAND_ASSIGN

mpiObjectMapBitCountMAX

Declaration

```
#define   mpiObjectMapBitCountMAX(objectMap)
                                  (sizeof(objectMap) * 8)
```

Required Header stdmpi.h

Description **ObjectMapBitCountMAX** calculates the maximum bit count for the specified object ***objectMap***.

See Also

mpiObjectMapBitGET

Declaration

```
#define mpiObjectMapBitGET(objectMap,bit)(((objectMap) &  (0x1 << (bit))) ? 1 : 0)
```

Required Header

stdmpi.h

Description

ObjectMapBitGET gets the bit number's state for the specified object ***map***.

See Also

mpiObjectMapBitSET

mpiObjectMapBitSET

Declaration

```
#define  mpiObjectMapBitSET(objectMap,bit,value) \
                              (((value) == 0) \
                          ? ((objectMap) &= ~(0x1 << (bit))) \
                          : ((objectMap) |=  (0x1 << (bit))))
```

Required Header stdmpi.h

Description **ObjectMapBitSET** sets object map's specified ***bit*** number to the specified ***value***.

See Also mpiObjectMapBitGET

mpiObjectMapCLEAR

Declaration

```
#define mpiObjectMapCLEAR(objectMap)  ((objectmap) = 0x0)
```

Required Header stdmpi.h

Description **ObjectMapCLEAR** sets the object ***map*** to zero.

See Also

mpiObjectMapCOMPLEMENT

Declaration

```
#define mpiObjectMapCOMPLEMENT(objectMap)
                                ((objectMap) = ~(objecMap))
```

Required Header

stdmpi.h

Description

ObjectMapCOMPLEMENT performs bitwise complement to the object map and assigns the result to ***objectMap***.

See Also

mpiObjectMapIS_CLEAR

Declaration

```
#define mpiObjectMapIS_CLEAR(objectMap) ((objectMap) == 0x0)
```

Required Header stdmpi.h

Description **ObjectMapIS_CLEAR** compares the map to 0x0. If the map is zero, it returns a non-zero value.

See Also

mpiObjectMapIS_EQUAL

Declaration

```
#define   mpiObjectMapIS_EQUAL(map1,map2)       ((map1) == (map2))
```

Required Header

stdmpi.h

Description

ObjectMapIS_EQUAL compares map1 to map2. If the maps are equal, it returns a non-zero value.

See Also

mpiObjectMapIS_VALID

Declaration

```
#define  mpiObjectMapIS_VALID(objectMap,count) \
                (((count) >= (sizeof(objectMap) * 8)) || \
                (((~((0x1 << (count)) - 1)) & (objectMap)) == 0))
```

Required Header stdmpi.h

Description **ObjectMapIS_VALID** checks if the map is within the count range. If the map is valid, it returns a non-zero value.

See Also

mpiObjectMapMAX

Declaration

```
#define mpiObjectMapMAX(objectMap,count)
                        ((objectMap) = (0x1 << (count)) - 1)
```

Required Header

stdmpi.h

Description

ObjectMapMAX calculates the maximum object ***objectMap*** with the specified ***count***.

See Also

mpiObjectMapOR_ASSIGN

Declaration

```
#define    mpiObjectMapOR_ASSIGN(dst,src)    ((dst) |= (src))
```

Required Header stdmpi.h

Description Bitwise ORs the dst object map with the ***src*** object map and assigns the result to ***dst***.

See Also mpiObjectMapASSIGN