Portable Systems Group

Interlocked Support Routines Specification

Author: *David N. Cutler*

Original Draft 1.0, June 7, 1989Revision 1.1, June 8, 1989Revision 1.2, July 15, 1989Revision 1.3, January 15, 1990
Revision 1.4, June 9, 1990

1. Introduction

This specification describes the interlocked support routines that are available for general use within the **Windows NT** executive. These routines can be used to provide properly synchronized access to nonpaged shared variables in a multiprocessor system.

These routines execute in kernel mode, raise IRQL to the highest level, synchronize with other processors using a spin lock, perform their operation, release the spin lock, and lower IRQL to its original value.

These routines cannot operate on paged data and are noninterruptable.

1.1 Interlocked Add Functions

Interlock add functions are provided to add unsigned short, long, and large integer values.

The addition is performed using a lock sequence so that access to the *Addend* variable is synchronized in a multiprocessor system.

1.1.1 Interlocked Add Unsigned Large Integer

An interlocked add operation can be performed on an integer of type **ULARGE_INTEGER** with the **ExInterlockedAddUlargeInteger** function:

ULARGE_INTEGER
ExInterlockedAddUlargeInteger (
    IN PULARGE_INTEGER *Addend*,
    IN ULARGE_INTEGER *Increment*,
    IN PKSPIN_LOCK *Lock*
    );

Parameters:

Addend - A pointer to a variable whose value is to be adjusted by the specified *Increment* value.

Increment - The increment value to be added to the specified *Addend* variable.

Lock - A pointer to a spin lock to be used to synchronize access to the *Addend* variable.

This function performs an interlocked add of an *Increment* value to an *Addend* variable, and stores the result in the *Addend* variable. The initial value of the *Addend* variable is returned as the function value.

1.1.2 Interlocked Add Unsigned Long

An interlocked add operation can be performed on an integer of type **ULONG** with the **ExInterlockedAddUlong** function:

ULONG
ExInterlockedAddUlong (
    IN PULONG *Addend*,
    IN ULONG *Increment*,
    IN PKSPIN_LOCK *Lock*
    );

Parameters:

Addend - A pointer to a variable whose value is to be adjusted by the specified *Increment* value.

Increment - The increment value to be added to the specified *Addend* variable.

Lock - A pointer to a spin lock to be used to synchronize access to the *Addend* variable.

This function performs an interlocked add of an *Increment* value to an *Addend* variable, and stores the result in the *Addend* variable. The initial value of the *Addend* variable is returned as the function value.

1.2.3 Interlocked Add Unsigned Short

An interlocked add operation can be performed on an integer of type **USHORT** with the **ExInterlockedAddUshort** function:

USHORT
ExInterlockedAddUshort (
    IN PUSHORT *Addend*,
    IN USHORT *Increment*,
    IN PKSPIN_LOCK *Lock*
    );

Parameters:

Addend - A pointer to a variable whose value is to be adjusted by the specified *Increment* value.

Increment - The increment value to be added to the specified *Addend* variable.

Lock - A pointer to a spin lock to be used to synchronize access to the *Addend* variable.

This function performs an interlocked add of an *Increment* value to an *Addend* variable, and stores the result in the *Addend* variable. The initial value of the *Addend* variable is returned as the function value.

1.2 Interlocked Doubly Linked List Functions

Interlocked functions are provided to insert at the head of a doubly linked list, to insert at the tail of a doubly linked list, and to remove from the head of a doubly linked list. The list head for an interlocked doubly linked list can be initialized with the standard **InitializeListHead** function.

1.2.1 Interlocked Insert Head Doubly Linked List

An entry can be inserted at the head of an interlocked doubly linked list with the **ExInterlockedInsertHeadList** function:

VOID
ExInterlockedInsertHeadList (
IN PLIST_ENTRY *ListHead*,
IN PLIST_ENTRY *ListEntry*,
IN PKSPIN_LOCK *Lock*
);

Parameters:

ListHead - A pointer to the head of the doubly linked list into which an entry is to be inserted.

ListEntry - A pointer to the list entry to be inserted at the head of the list.

Lock - A pointer to a spin lock to be used to synchronize access to the interlocked list.

This function inserts an entry at the head of a doubly linked list so that access to the list is synchronized in a multiprocessor system.

1.2.2 Interlocked Insert Tail Doubly Linked List

An entry can be inserted at the tail of an interlocked doubly linked list with the **ExInterlockedInsertTailList** function:

VOID
ExInterlockedInsertTailList (
IN PLIST_ENTRY *ListHead*,
IN PLIST_ENTRY *ListEntry*,
IN PKSPIN_LOCK *Lock*
);

Parameters:

ListHead - A pointer to the head of the doubly linked list into which an entry is to be inserted.

ListEntry - A pointer to the list entry to be inserted at the tail of the list.

Lock - A pointer to a spin lock to be used to synchronize access to the interlocked list.

This function inserts an entry at the tail of a doubly linked list so that access to the list is synchronized in a multiprocessor system.

1.2.3 Interlocked Remove Doubly Linked List

An entry can be removed from the head of an interlocked doubly linked list with the **ExInterlockedRemoveHeadList** function:

PLIST_ENTRY
ExInterlockedRemoveHeadList (
IN PLIST_ENTRY *ListHead*,
IN PKSPIN_LOCK *Lock*
);

Parameters:

ListHead - A pointer to the head of the doubly linked list from which an entry is to be removed.

Lock - A pointer to a spin lock to be used to synchronize access to the interlocked list.

This function removes an entry from the head of a doubly linked list so that access to the list is synchronized in a multiprocessor system. If there are no entries in the list, then a value of NULL is returned. Otherwise, the address of the entry that is removed from the list is returned as the function value.

1.3 Interlocked Singly Linked List Functions

Interlocked functions are provided to insert and remove from the head of a singly linked list. The list head for an interlocked singly linked list can be initialized by simply placing a value of NULL in the link pointer.

1.3.1 Interlocked Insert Head Singly Linked List

An entry can be inserted at the head of an interlocked singly linked list with the **ExInterlockedPushEntryList** function:

VOID
ExInterlockedPushEntryList (
    IN PSINGLE_LIST_ENTRY *ListHead*,
    IN PSINGLE_LIST_ENTRY *ListEntry*,
    IN PKSPIN_LOCK *Lock*
    );

Parameters:

ListHead - A pointer to the head of the singly linked list into which an entry is to be inserted.

ListEntry - A pointer to the list entry to be inserted at the head of the list.

Lock - A pointer to a spin lock to be used to synchronize access to the interlocked list.

This function inserts an entry at the head of a singly linked list so that access to the list is synchronized in a multiprocessor system.

1.3.2 Interlocked Remove Singly Linked List

An entry can be removed from the head of an interlocked singly linked list with the **ExInterlockedPopEntryList** function:

PSINGLE_LIST_ENTRY
ExInterlockedPopEntryList (
IN PSINGLE_LIST_ENTRY *ListHead*,
IN PKSPIN_LOCK *Lock*
);

Parameters:

ListHead - A pointer to the head of the singly linked list from which an entry is to be removed.

Lock - A pointer to a spin lock to be used to synchronize access to the interlocked list.

This function removes an entry from the head of a singly linked list so that access to the list is synchronized in a multiprocessor system. If there are no entries in the list, then a value of NULL is returned. Otherwise, the address of the entry that is removed from the list is returned as the function value.

Revision History:

Original Draft 1.0, June 7, 1989

Revision 1.1, June 8,1989

1. Added comments about nonpageability of data to the introduction.
2. Clarified functions as operating on singly or doubly linked lists.
3. Added section on the manipulation of singly linked interlocked lists.

Revision 1.2, July 15, 1989

1. Add interlocked add short routine.

Revision 1.3, January 15, 1990

2. Remove restriction that page faults cannot be taken during interlocked sequences.

Revision 1.4, June 9, 1990

1. Change the operation of the interlocked add long and short functions to be unsigned and change the names as appropriate.
2. Add a function to perform an interlocked add unsigned large integer operation.
3. Add text to warn users of these routines that page faults can not be tolerated.