xPC Target Function Library Quick Reference Sheet

Note: Header file `xpctarget.h` contains function prototypes for all xPC Target functions

Data Structures

Structure	Structure Description	Members	Type	Member Description
`xpcPCIDevice`	Defines PCI configuration space	`BaseAddress[6]`	`uint32_T`	Physical base address assigned by PCI BIOS
		`VirtAddress[6]`	`uint32_T`	Virtual address (see xpcReserveMemoryRegion)
		`Length[6]`	`uint32_T`	Length of region in bytes
		`AddressSpaceIndicator[6]`	`uint16_T`	0: Memory-mapped, 1: I/O port mapped
		`MemoryType[6]`	`uint16_T`	0: 32-bit, 1: Locate below 1MB, 2: 64-bit
		`Prefetchable[6]`	`uint16_T`	Indicates if memory is prefetchable
		`InterruptLine`	`uint16_T`	Contains the assigned interrupt line, 0-15
		`VendorId`	`uint16_T`	Vendor ID
		`DeviceId`	`uint16_T`	Device ID
		`SubDeviceId`	`uint16_T`	Sub-device ID
		`SubVendorId`	`uint16_T`	Sub-vendor ID
`xpcTime`	Holds time value in nanoseconds	`U64.NanoSecondsLo`	`uint32_T`	Bottom 32 bits of 64-bit value in nanoseconds
		`U64.NanoSecondsHi`	`uint32_T`	Top 32 bits of 64-bit value in nanoseconds

Functions

Function Name	Description	Returns	Arguments
`xpcInpB`	Returns value of byte input port	`uint8_T value`	`uint16_T port`
`xpcInpW`	Returns value of word input port	`uint16_T value`	`uint16_T port`
`xpcInpDW`	Returns value of double word input port	`uint32_T value`	`uint16_T port`
`xpcOutpB`	Writes to byte output port	`void`	`uint16_T port, uint8_T value`
`xpcOutpW`	Writes to word output port	`void`	`uint16_T port, uint16_T value`
`xpcOutpDW`	Writes to double word output port	`void`	`uint16_T port, uint32_T value`
`xpcGetPCIDeviceInfo`	Get PCI device information	`int32_T 0: pass 1: fail`	`uint16_T vendorId, uint16_T deviceId, uint16_T subVendorId, uint16_T subDeviceId, uint16_T bus, uint16_T slot, xpcPCIDevice *pciInfo`
`xpcShowPCIDeviceInfo`	Display PCI device information	`void`	`xpcPCIDevice *pciInfo`
`xpcAllocPhysicalMemory`	Allocate physical memory	`void * physical`	`uint32_T numBytes`
`xpcFreePhysicalMemory`	Free physical memory	`void`	`const void *physical`
`xpcReserveMemoryRegion`	Map physical memory to virtual	`void * virtual`	`const void *physical, uint32_T numBytes, uint32_T access`
`xpcGetElapsedTime`	Returns time since system boot	`real_T seconds`	`xpcTime *upTime`
`xpcSubtractTime`	Returns difference between two times	`real_T seconds`	`xpcTime *time, const xpcTime *time2, const xpcTime *time1`
`xpcBusyWait`	Wait for specified length of time	`void`	`real_T seconds`
`xpcIsModelInit`	Returns target model load state; true: while application loads, false: start application execution	`boolean_T load`	`void`

Interrupt Hook Functions

Function Name	Description	Returns	Arguments	PCI	ISA
`PreHookFunction`	ISR function to be called when board issues interrupt request	`int XPC_RUN_ISR XPC_DROP_ISR`	`xpcPCIDevice *pciInfo`	All members of `xpcPCIDevice` except `VirtAddress` are provided.	Only `BaseAddress[0]` of `xpcPCIDevice` is provided.
`PostHookFunction`	Runs after return from function call on interrupt, before model execution	`void`	`xpcPCIDevice *pciInfo`		
`StartFunction`	Called before model begins execution to enable interrupts	`void`	`xpcPCIDevice *pciInfo`		
`StopFunction`	Called after model termination to disable interrupts	`void`	`xpcPCIDevice *pciInfo`		

Macros

#define	Description
`XPC_NO_VEND_DEV`	Use when no VendorId, DeviceId, subVendorId or subDeviceId is required
`XPC_NO_BUS_SLOT`	Use when not specifying PCI bus or slot
`XPC_RT_PG_USERREADWRITE`	Read/write access to memory region
`XPC_RUN_ISR`	Hook function return if interrupt occurred
`XPC_DROP_ISR`	Hook function return if no interrupt occurred