Confidential
For PROTRULY Only

SDK6 API: Kernel Abstraction Layer

(AmbaKAL)

Version 1.8

June 5, 2015

Confidentiality Notice:

Copyright © 2015 Ambarella, Inc.

The contents of this document are proprietary and confidential information of Ambarella, Inc.

The material in this document is for information only. Ambarella assumes no responsibility for errors or omissions and reserves the right to change, without notice, product specifications, operating characteristics, packaging, ordering, etc. Ambarella assumes no liability for damage resulting from the use of information contained in this document. All brands, product names, and company names are trademarks of their respective owners.

Confidential - For PROTRUST Only

US
3101 Jay Street
Ste.110
Santa Clara, CA 95054, USA
Phone: +1.408.734.8888
Fax: +1.408.734.0788

Taiwan
Suite C1, No. 1, Li-Hsin Road 1
Science-Based Industrial Park
Hsinchu 30078, Taiwan
Phone: +886.3.666.8828
Fax: +886.3.666.1282

Hong Kong
Unit A&B, 18/F, Spectrum Tower
53 Hung To Road
Kwun Tong, Kowloon
Phone: +85.2.2806.8711
Fax: +85.2.2806.8722

Japan - Yokohama
Shin-Yokohama Business Center Bldg. 5th Floor
3-2-6 Shin-Yokohama, Kohoku-ku,
Yokohama, Kanagawa, 222-0033, Japan
Phone: +81.45.548.6150
Fax: +81.45.548.6151

Korea
6 Floor, Hanwon-Bldg.
Sunae-Dong, 6-1, Bundang-Gu
SeongNam-City, Kyunggi-Do
Republic of Korea 463-825
Phone: +031.717.2780
Fax: +031.717.2782

China - Shenzhen
Unit E, 5th Floor
No. 2 Finance Base
8 Ke Fa Road
Shenzhen, 518057, China
Phone: +86.755.3301.0366
Fax: +86.755.3301.0966

China - Shanghai
9th Floor, Park Center
1088 Fangdian Road, Pudong New District
Shanghai 201204, China
Phone: +86.21.6088.0608
Fax: +86.21.6088.0366

I Contents

Confidential For PROTRULY Only

II Preface

This document provides technical detail using a set of consistent typographical conventions to help the user differentiate key concepts at a glance.

Conventions include:

Example	Description
AmbaGuiGen, **DirectUSB** **Save, File > Save** **Power, Reset, Home**	Software names GUI commands and command sequences Computer / Hardware buttons
Flash_IO_control **da**, **status**, **enable**	Register names and register fields. For example, **Flash_IO_control** is the register for global control of Flash I/O, and bit 17 (**da)** is used for DMA acknowledgement.
GPIO81, **CLK_AU**	Hardware external pins
VIL, VIH, VOL, VOH	Hardware pin parameters
INT_O, RXDATA_I	Hardware pin signals
amb_performance_t ***amb_operating_mode_t*** **amb_set_operating_mode()**	API details (e.g., functions, structures, and type definitions)

`yes`
`/usr/local/bin`
`make`
	User entries into software dialogues and GUI windows
	File names and paths
	Command line scripting and Code

Table II-1. Typographical Conventions for Technical Documents.

Additional Ambarella typographical conventions include:

- Acronyms are given in UPPER CASE using the default font (e.g., AHB, ARM11 and DDRIO).
- Names of Ambarella documents and publicly available standards, specifications, and databooks appear in *italic* type.

Confidential
For PROTRUY Only

1 Overview

1.1 Overview: Introduction

This document provides the Ambarella Kernel Abstraction Layer (AmbaKAL) APIs. The document is organized as follows:

- (Chapter 1) Overview
- (Chapter 2) Kernel Abstraction Layer API

1.2 Overview: AmbaKAL

The AmbaKAL API abstracts kernel-level detail to simplify the creation, management and timing of software tasks and applications. Straightforward functions (e.g. Create, Delete, Give and Query, Take) streamline program creation, prioritizing and resource sharing. The AmbaKAL API also abstracts the definitions of key process-control blocks and provides pointers to simplify access and management.

For SMP (Symmetric Multi-Processors) machines, AmbaKAL allows user to lock tasks/timers in certain cores. Furthermore, it also provides the method to prevent concurrent core access to the shared data. Please reference CriticalSection series API for more detail.

For Ambarella Confidential PROTRULY Only

1.3 Overview: Scope of Document

This document focuses strictly on the A9 Kernel Abstraction Layer API. Users of this document are assumed to be familiar with the A9 chip hardware, system capabilities, software architecture and reference applications. The reader is referred to the following for a background overview:

- The *AMBARELLA A9X chip datasheet* provides hardware pin and package details including a feature list with descriptions of chip performance, brief interface descriptions, a complete power-on configuration table and electrical characteristics.
- The *AMBARELLA A9 Hardware Programming Reference Manual* is the primary resource for programming peripheral drivers. It lists software-programmable registers accessible from CPU cores, including detailed information on each field of a register. It also provides overviews of the system memory map, power-on configuration options, and ARM interrupts.
- *AMBARELLA A9: System Hardware* covers power-on timing and sequencing. It provides pin connection details including guidance for unused interfaces and PCB layout.

2 Kernel Abstraction Layer API

2.1 KAL: Overview

This chapter details the AmbaKAL API commands. The chapter is organized as follows:

- (Section 2.2) KAL: Process Control Blocks
- (Section 2.3) KAL: List of Functions

2.2 KAL: Process Control Blocks

The AmbaKAL API abstracts key process-control block definitions and provides pointers that can be used to access them. AmbaKAL commands are organized according to control the block.

Access Pointer	Control Block Description
AMBA_KAL_TASK_t	Software Task Control Block
AMBA_KAL_SEM_t	Semaphore Control Block
AMBA_KAL_MUTEX_t	Mutex Control Block
AMBA_KAL_EVENT_FLAG_t	Event Flag Group Control Block
AMBA_KAL_MSG_QUEUE_t	Message Queue Control Block
AMBA_KAL_MSG_TIMER_t	Timer Control Block
AMBA_KAL_BYTE_POOL_t	Byte Pool Control Block
AMBA_KAL_BLOCK_POOL_t	Block Pool Control Block

Table 2-1. Process Control Block Definitions.

Confidential
For PROPRIETARY Only

2.3 KAL: List of Functions

/*Task*/

- AmbaKAL_TaskCreate
- AmbaKAL_TaskDelete
- AmbaKAL_TaskTerminate
- AmbaKAL_TaskReset
- AmbaKAL_TaskSuspend
- AmbaKAL_TaskResume
- AmbaKAL_TaskSleep
- AmbaKAL_TaskQuery
- AmbaKAL_TaskChangePriority

- AmbaKAL_TaskIdentify
- AmbaKAL_TaskFpuEnable
- AmbaKAL_TaskFpuDisable

/*User Specific*/

- AmbaKAL_TaskUserNotify
- AmbaKAL_TaskUserValueGet
- AmbaKAL_TaskUserValueSet
- AmbaKAL_RegisterStackErrorHandler
- AmbaKAL_IsInISR

/*Critical Section*/

- AmbaKAL_EnterCriticalSection
- AmbaKAL_ExitCriticalSection

/*Semaphore*/

- AmbaKAL_SemCreate
- AmbaKAL_SemDelete
- AmbaKAL_SemTake
- AmbaKAL_SemGive
- AmbaKAL_SemQuery

/*Mutex*/

- AmbaKAL_MutexCreate
- AmbaKAL_MutexDelete
- AmbaKAL_MutexTake
- AmbaKAL_MutexGive

/*EventGlags*/

- AmbaKAL_EventFlagCreate
- AmbaKAL_EventFlagDelete
- AmbaKAL_EventFlagTake
- AmbaKAL_EventFlagGive
- AmbaKAL_EventFlagClear
- AmbaKAL_EventFlagQuery

/*Message Queue*/

- AmbaKAL_MsgQueueCreate
- AmbaKAL_MsgQueueDelete
- AmbaKAL_MsgQueueFlush
- AmbaKAL_MsgQueueReceive
- AmbaKAL_MsgQueueSend

Confidential
For PROTRULY Only

/*Timer*/

- AmbaKAL_TimerCreate
- AmbaKAL_TimerDelete
- AmbaKAL_TimerStart
- AmbaKAL_TimerStop
- AmbaKAL_TimerChange
- AmbaKAL_GetTickCount

/*Memory pool*/

- AmbaKAL_MemAllocate
- AmbaKAL_MemFree
- AmbaKAL_BytePoolCreate
- AmbaKAL_BytePoolDelete
- AmbaKAL_BytePoolAllocate
- AmbaKAL_BytePoolFree
- AmbaKAL_BytePoolInfoGet
- AmbaKAL_BlockPoolCreate
- AmbaKAL_BlockPoolDelete
- AmbaKAL_BlockPoolAllocate
- AmbaKAL_BlockPoolFree

/*SMP*/

- AmbaKAL_TaskSmpCoreExclusionSet
- AmbaKAL_TaskSmpCoreExclusionGet
- AmbaKAL_TaskSmpCurCoreGet
- AmbaKAL_TimerSmpCoreExclusionSet
- AmbaKAL_TimerSmpCoreExclusionGet

For PROTRULY Only
Confidential

2.4 KAL: Task Management Function

This section explains the task management functions that provide direct control of task states and reference to the task states.

Confidential
For PROTRULY Only

2.4.1 AmbaKAL_TaskCreate

API Syntax:

AmbaKAL_TaskCreate (AMBA_KAL_TASK_t *pTask, char *pTaskName, UINT32 Priority, void (*Entry-Function)(UINT32), UINT32 EntryArg, void *pStackBase, UINT32 StackByteSize, UINT32 AutoStart)

Function Description:

- This function is used to generate a software task with a defined configuration

Parameters:

Type	Parameter	Description
AMBA_KAL_ TASK_t	***pTask**	Pointer to a Software Task Control Block
char	***pTaskName**	Pointer to the Software Task name
UINT32	**Priority**	Priority of the Software Task: 0: **AMBA_KAL_TASK_HIGHEST_PRIORITY** MAX -1: **AMBA_KAL_TASK_LOWEST_PRIORITY** where the maximum number of priorities **TX_MAX_PRIOR-ITES** (MAX) = 256
void	**(*EntryFunction)**	Entry function for the Software Task
UINT32	**EntryArg**	32-bit input argument for the Software Task Entry function
void	***pStackBase**	Pointer to the stack area
UINT32	**StackByteSize**	Stack size in bytes. Valid range is 0 up to system memory with 0xFFFFFFFF maximum. Number bytes in the stack memory area. The thread stack area must be large enough to handle its worst-case function call nesting and local variable usage.
UINT32	**AutoStart**	Automatic start selection Specifies whether the thread starts immediately or is placed in a suspended state. Legal options are: 0x0: **AMBA_KAL_DO_NOT_START** 0x1: **AMBA_KAL_AUTO_START** If **AMBA_KAL_DO_NOT_START** is specified, the application must later call **AmbaKAL_TaskResume** for the task to run.

*Table 2-2. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskCreate()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-3. Returns for Kernel AmbaKAL API **AmbaKAL_TaskCreate()**.*

For PROOFREADING Only Confidential

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.4.2 AmbaKAL_TaskDelete

Kernel

API Syntax:

AmbaKAL_TaskDelete (AMBA_KAL_TASK_t *pTask)

Function Description:

- This function is used to delete a specified software task.

Parameters:

Type	Parameter	Description
AMBA_KAL_TASK_t	***pTask**	Pointer to the Software Task Control Block

*Table 2-4. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskDelete()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-5. Returns for Kernel AmbaKAL API **AmbaKAL_TaskDelete()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.4.3 AmbaKAL_TaskTerminate

Kernel

API Syntax:

AmbaKAL_TaskTerminate (AMBA_KAL_TASK_t *pTask)

Function Description:

- This function is used to terminate the operation of a software task.

Parameters:

Type	Parameter	Description
AMBA_KAL_TASK_t	***pTask**	Pointer to the Software Task Control Block

*Table 2-6. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskTerminate()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-7. Returns for Kernel AmbaKAL API **AmbaKAL_TaskTerminate()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.4.4 AmbaKAL_TaskReset

API Syntax:

AmbaKAL_TaskReset(AMBA_KAL_TASK_t *pTask)

Function Description:

- This function is used to reset a previously terminated software task so that it will execute at its entry point. The task must be in a terminated state for it to be reset.

Parameters:

Type	Parameter	Description
AMBA_KAL_TASK_t	***pTask**	Pointer to the Software Task Control Block

*Table 2-8. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskReset()***

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-9. Returns for Kernel AmbaKAL API **AmbaKAL_TaskReset()**.*

Example:

```
AMBA_KAL_TASK_t  MyTask;

AmbaKAL_TaskReset(&MyTask);  /* Reset MyTask which is in terminated state */
```

See Also:

None

Confidential
For PROTRULY Only

2.4.5 AmbaKAL_TaskSuspend

API Syntax:

AmbaKAL_TaskSuspend (AMBA_KAL_TASK_t *pTask)

Function Description:

- This function is used to suspend the operations of a software task without specifying a resume time.

Parameters:

Type	Parameter	Description
AMBA_KAL_TASK_t	***pTask**	Pointer to the Software Task Control Block

*Table 2-10. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskSuspend()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-11. Returns for Kernel AmbaKAL API **AmbaKAL_TaskSuspend()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.4.6 AmbaKAL_TaskResume

API Syntax:

AmbaKAL_TaskResume (AMBA_KAL_TASK_t *pTask)

Function Description:

- This function is used to resume a software task that has been placed in a suspended state.

Parameters:

Type	Parameter	Description
AMBA_KAL_TASK_t	***pTask**	Pointer to the Software Task Control Block

*Table 2-12. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskResume()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-13. Returns for Kernel AmbaKAL API **AmbaKAL_TaskResume()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.4.7 AmbaKAL_TaskSleep

API Syntax:

AmbaKAL_TaskSleep (int TimeValue)

Function Description:

- This function is used to suspend a software task for a defined time period, after which the task will resume.

Parameters:

Type	Parameter	Description
int	**TimeValue**	The number of timer ticks to suspend the task. Valid range is 0x00000000 to 0xFFFFFFFF. If 0 is specified, the service returns immediately.

*Table 2-14. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskSleep()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-15. Returns for Kernel AmbaKAL API **AmbaKAL_TaskSleep()**.*

Example:

None

See Also:

None

Confidential
FOR PROTRULY Only

2.4.8 AmbaKAL_TaskQuery

API Syntax:

AmbaKAL_TaskQuery(AMBA_KAL_TASK_t *pTask, UINT32 *pCurState)

Function Description:

- This function retrieves the state of a specified software task.

Parameters:

Type	Parameter	Description
AMBA_KAL_TASK_t	***pTask**	Pointer to the Software Task Control Block
UINT32	***pCurState**	Pointer to the current state of Software task

*Table 2-16. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskQuery()***

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-17. Returns for Kernel AmbaKAL API **AmbaKAL_TaskQuery()**.*

Example:

```
AMBA_KAL_TASK_t  MyTask;
UINT32 CurState;

AmbaKAL_TaskQuery (&MyTask, &CurState);
```

See Also:

None

Confidential
For PROTRULY Only

2.4.9 AmbaKAL_TaskChangePriority

API Syntax:

AmbaKAL_TaskChangePriority (AMBA_KAL_TASK_t *pTask, UINT32 NewPriority, UINT32 *pOldPriority)

Function Description:

- This function is used to change the priority level of a software task.

Parameters:

Type	Parameter	Description
AMBA_KAL_TASK_t	***pTask**	Pointer to the Software Task Control Block
UINT32	**NewPriority**	New priority. See Section 2.4.1 “AmbaKAL_TaskCreate” for **Priority** definition.
UINT32	***pOldPriority**	Pointer to the previous priority

*Table 2-18. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskChangePriority()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-19. Returns for Kernel AmbaKAL API **AmbaKAL_TaskChangePriority()**.*

Example:

None

See Also:

AmbaKAL_TaskCreate()

Confidential
For PROTRULY Only

2.4.10 AmbaKAL_TaskIdentify

API Syntax:

AmbaKAL_TaskIdentify (void)

Function Description:

- This function returns a pointer to the currently executing thread. If no thread is executing, this service returns a null pointer.

Parameters:

None

Returns:

Return	Description
AMBA_KAL_TASK_t	The thread pointer
NULL	No thread is executing

*Table 2-20. Returns for Kernel AmbaKAL API **AmbaKAL_TaskIdentify()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.4.11 AmbaKAL_TaskFpuEnable

API Syntax:

AmbaKAL_TaskFpuEnable (void)

Function Description:

- This function is used to modify the kernel that floating point operations will be used by the currently executing thread. The kernel needs to take care of the additional registers saving/restoring during context switches. This FPU support is enabled by default after each task is created.

Parameters:

None

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-21. Returns for Kernel AmbaKAL API **AmbaKAL_TaskFpuEnable()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.4.12 AmbaKAL_TaskFpuDisable

API Syntax:

AmbaKAL_TaskFpuDisable (void)

Function Description:

- This function is used to disable FPU support for the currently executing thread. It might improve a little performance but it is very dangerous if the user is not certain if there is any FPU operation within the task or not.

Parameters:

None

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-22. Returns for Kernel AmbaKAL API **AmbaKAL_TaskFpuDisable()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.5 KAL: User Specific

This section explains the user specific functions, which provide applications with ability for users to set/get user value or register callback functions for checking thread status.

Confidential
For PROTRULY Only

2.5.1 AmbaKAL_TaskUserNotify

API Syntax:

AmbaKAL_TaskUserNotify (AMBA_KAL_TASK_t *pTask, void (*UserFunction)(AMBA_KAL_TASK_t *, UINT32)

Function Description:

- This function is used to register a notification callback function that is called whenever the system enters or exits the specified thread.

Parameters:

Type	Parameter	Description
AMBA_KAL_TASK_t	***pTask**	Pointer to the Software Task Control Block
void	**(*UserFunction)**	Pointer to the notification callback function

*Table 2-23. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskUserNotify()**.*

Returns:

Return	Description
0	Success
0x0E	Failure

*Table 2-24. Returns for Kernel AmbaKAL API **AmbaKAL_TaskUserNotify()**.*

Example:

None

See Also:

None

Confidential
For PROPERLY Only

2.5.2 AmbaKAL_TaskUserValueGet

API Syntax:

AmbaKAL_TaskUserValueGet (AMBA_KAL_TASK_t *pTask, UINT32 *pUserValue)

Function Description:

- This function is used to get the user value.

Parameters:

Type	Parameter	Description
AMBA_KAL_TASK_t	***pTask**	Pointer to the Software Task Control Block
UINT32	***pUserValue**	Pointer to the buffer of the user value

*Table 2-25. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskUserValueGet()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-26. Returns for Kernel AmbaKAL API **AmbaKAL_TaskUserValueGet()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.5.3 AmbaKAL_TaskUserValueSet

API Syntax:

AmbaKAL_TaskUserValueSet (AMBA_KAL_TASK_t *pTask, UINT32 UserValue)

Function Description:

- This function is used to set the user value.

Parameters:

Type	Parameter	Description
AMBA_KAL_TASK_t	***pTask**	Pointer to the Software Task Control Block
UINT32	**UserValue**	User value

*Table 2-27. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskUserValueSet()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-28. Returns for Kernel AmbaKAL API **AmbaKAL_TaskUserValueSet()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.5.4 AmbaKAL_RegisterStackErrorHandler

API Syntax:

AmbaKAL_RegisterStackErrorHandler (void (*StackErrorHandler)(AMBA_KAL_TASK_t *))

Function Description:

- This function is used to register the software task stack error notification callback.

Parameters:

Type	Parameter	Description
void	**(*StackErrorHandler)**	Pointer to the software task stack error notification handler

*Table 2-29. Parameters for Kernel AmbaKAL API **AmbaKAL_RegisterStackErrorHandler()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-30. Returns for Kernel AmbaKAL API **AmbaKAL_RegisterStackErrorHandler()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.5.5 AmbaKAL_IsInISR

API Syntax:

AmbaKAL_IsInISR (void)

Function Description:

- This function is used to check if current system state is in Interrupt Service Routine (ISR) or not.

Parameters:

None

Returns:

Return	Description
0	Not in ISR
-1	In ISR

*Table 2-31. Returns for Kernel AmbaKAL API **AmbaKAL_IsInISR()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.6 KAL: Critical Section

This section explains the critical section, which is a piece of code that accesses a shared resource that must not be concurrently accessed by more than one entry point.

Confidential
For PROTRULY Only

2.6.1 AmbaKAL_EnterCriticalSection

API Syntax:

AmbaKAL_EnterCriticalSection (void)

Function Description:

- This function is used to request the ownership of the critical section. If one core obtains ownership of the critical section, it will be non-interruptible and all the other cores will be blocked until the core relinquishes ownership.

Parameters:

None

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-32. Returns for Kernel AmbaKAL API **AmbaKAL_EnterCriticalSection()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.6.2 AmbaKAL_ExitCriticalSection

API Syntax:

AmbaKAL_ExitCriticalSection (void)

Function Description:

- This function is used to release the ownership of the critical section.

Parameters:

None

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-33. Returns for Kernel AmbaKAL API **AmbaKAL_ExitCriticalSection()**.*

Example:

None

See Also:

None

For Confidential PROTRULY Only

2.7 KAL: Counting Semaphores

This section explains the semaphore, which is an object used for mutual exclusion and synchronization. A semaphore indicates availability and the number of unused resources by a resource count.

Confidential
For PROTRULY Only

2.7.1 AmbaKAL_SemCreate

API Syntax:

AmbaKAL_SemCreate (AMBA_KAL_SEM_t *pSem, UINT32 InitCount)

Function Description:

- This function is used to generate a counting semaphore with a defined configuration.

Parameters:

Type	Parameter	Description
AMBA_KAL_SEM_t	***pSem**	Pointer to a Semaphore Control Block
UINT32	**InitCount**	Initial count for this semaphore. Valid range is 0x00000000 to 0xFFFFFFFF.

*Table 2-34. Parameters for Kernel AmbaKAL API **AmbaKAL_SemCreate()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-35. Returns for Kernel AmbaKAL API **AmbaKAL_SemCreate()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.7.2 AmbaKAL_SemDelete

API Syntax:

AmbaKAL_SemDelete (AMBA_KAL_SEM_t *pSem)

Function Description:

- This function is used to delete a specified counting semaphore.

Parameters:

Type	Parameter	Description
AMBA_KAL_SEM_t	***pSem**	Pointer to a Semaphore Control Block

*Table 2-36. Parameters for Kernel AmbaKAL API **AmbaKAL_SemDelete()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-37. Returns for Kernel AmbaKAL API **AmbaKAL_SemDelete()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.7.3 AmbaKAL_SemTake

API Syntax:

AmbaKAL_SemTake (AMBA_KAL_SEM_t *pSem, UINT32 Timeout)

Function Description:

- This function is used to take (i.e., decrement) a specified counting semaphore.

Parameters:

Type	Parameter	Description
AMBA_KAL_SEM_t	***pSem**	Pointer to the Semaphore Control Block
UINT32	**Timeout**	Timeout value in milliseconds (ms). Valid range is 0x00000000 to 0xFFFFFFFF.

*Table 2-38. Parameters for Kernel AmbaKAL API **AmbaKAL_SemTake()**.*

Returns:

Return	Description
0	Success
- 1	Failure

*Table 2-39. Returns for Kernel AmbaKAL API **AmbaKAL_SemTake()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.7.4 AmbaKAL_SemGive

API Syntax:

AmbaKAL_SemGive (AMBA_KAL_SEM_t *pSem)

Function Description:

- This function is used to give (i.e., increment) a specified counting semaphore.

Parameters:

Type	Parameter	Description
AMBA_KAL_SEM_t	***pSem**	Pointer to the Semaphore Control Block

*Table 2-40. Parameters for Kernel AmbaKAL API **AmbaKAL_SemGive()**.*

Returns:

Return	Description
0	Success
- 1	Failure

*Table 2-41. Returns for Kernel AmbaKAL API **AmbaKAL_SemGive()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.7.5 AmbaKAL_SemQuery

API Syntax:

AmbaKAL_SemQuery (AMBA_KAL_SEM_t *pSem, UINT32 *pCurCount)

Function Description:

- This function is used to retrieve the current count of a counting semaphore.

Parameters:

Type	Parameter	Description
AMBA_KAL_SEM_t	***pSem**	Pointer to the Semaphore Control Block
UINT32	***pCurCount**	Pointer to the count of the current semaphore

*Table 2-42. Parameters for Kernel AmbaKAL API **AmbaKAL_SemQuery()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-43. Returns for Kernel AmbaKAL API **AmbaKAL_SemQuery()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.8 KAL: Mutexes

This section explains the mutex, which is an object used for mutual exclusion of a shared resource among tasks. It is basically a binary semaphore, which means that only one task can own a mutex at a time. A mutex has a locked and unlocked state. It also has a wait queue for tasks waiting to lock the mutex.

Confidential
For PROTRULY Only

2.8.1 AmbaKAL_MutexCreate

API Syntax:

AmbaKAL_MutexCreate (AMBA_KAL_MUTEX_t *pMutex)

Function Description:

- This function is used to generate a mutex object.

Parameters:

Type	Parameter	Description
AMBA_KAL_MUTEX_t	***pMutex**	Pointer to the Mutex Control Block

*Table 2-44. Parameters for Kernel AmbaKAL API **AmbaKAL_MutexCreate()**.*

Returns:

Return	Description
0	Success
- 1	Failure

*Table 2-45. Returns for Kernel AmbaKAL API **AmbaKAL_MutexCreate()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.8.2 AmbaKAL_MutexDelete

API Syntax:

AmbaKAL_MutexDelete (AMBA_KAL_MUTEX_t *pMutex)

Function Description:

- This function is used to delete a specified mutex object.

Parameters:

Type	Parameter	Description
AMBA_KAL_MUTEX_t	***pMutex**	Pointer to the Mutex Control Block

*Table 2-46. Parameters for Kernel AmbaKAL API **AmbaKAL_MutexDelete()**.*

Returns:

Return	Description
0	Success
- 1	Failure

*Table 2-47. Returns for Kernel AmbaKAL API **AmbaKAL_MutexDelete()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.8.3 AmbaKAL_MutexTake

API Syntax:

AmbaKAL_MutexTake (AMBA_KAL_MUTEX_t *pMutex, UINT32 Timeout)

Function Description:

- This function is used to take (i.e., decrement) a specified mutex object.

Parameters:

Type	Parameter	Description
AMBA_KAL_MUTEX_t	***pMutex**	Pointer to the Mutex Control Block
UINT32	**Timeout**	Timeout value in milliseconds (ms). Valid range is 0x00000000 to 0xFFFFFFFF.

*Table 2-48. Parameters for Kernel AmbaKAL API **AmbaKAL_MutexTake()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-49. Returns for Kernel AmbaKAL API **AmbaKAL_MutexTake()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.8.4 AmbaKAL_MutexGive

API Syntax:

AmbaKAL_MutexGive (AMBA_KAL_MUTEX_t *pMutex)

Function Description:

- This function is used to give (i.e., increment) a specified mutex object.

Parameters:

Type	Parameter	Description
AMBA_KAL_MUTEX_t	***pMutex**	Pointer to the Mutex Control Block

*Table 2-50. Parameters for Kernel AmbaKAL API **AmbaKAL_MutexGive()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-51. Returns for Kernel AmbaKAL API **AmbaKAL_MutexGive()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.9 KAL: EventFlags

This section explains the eventflag, which is a synchronization object that consists of multiple bits in a bit pattern where each bit represents an event.
An eventflag has an associated bit pattern expressing the state of its events, and a wait queue for tasks waiting on these events.

Confidential
For PROTRULY Only

2.9.1 AmbaKAL_EventFlagCreate

API Syntax:

AmbaKAL_EventFlagCreate (AMBA_KAL_EVENT_FLAG_t *pEventFlag)

Function Description:

- This function is used to generate a group of 32 event flags. All 32 event flags in the group are initialized to zero. Each event flag is represented by a single bit.

Parameters:

Type	Parameter	Description
AMBA_KAL_EVENT_FLAG_t	***pEventFlag**	Pointer to an Event Flag Group Control Block.

*Table 2-52. Parameters for Kernel AmbaKAL API **AmbaKAL_EventFlagCreate()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-53. Returns for Kernel AmbaKAL API **AmbaKAL_EventFlagCreate()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.9.2 AmbaKAL_EventFlagDelete

API Syntax:

AmbaKAL_EventFlagDelete (AMBA_KAL_EVENT_FLAG_t *pEventFlag)

Function Description:

- This function is used to delete a specified event flag group.

Parameters:

Type	Parameter	Description
AMBA_KAL_EVENT_FLAG_t	***pEventFlag**	Pointer to an Event Flag Group Control Block

*Table 2-54. Parameters for Kernel AmbaKAL API **AmbaKAL_EventFlagDelete()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-55. Returns for Kernel AmbaKAL API **AmbaKAL_EventFlagDelete()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.9.3 AmbaKAL_EventFlagTake

API Syntax:

AmbaKAL_EventFlagTake (AMBA_KAL_EVENT_FLAG_t *pEventFlag, UINT32 ReqFlags, UINT32 Option, UINT32 *pActualFlags, UINT32 Timeout)

Function Description:

- This function is used to take event flags from an event flag group.

Parameters:

Type	Parameter	Description
AMBA_KAL_EVENT_FLAG_t	***pEventFlag**	Pointer to an Event Flag Group Control Block
UINT32	**ReqFlags**	A bit pattern indicating which bits to check. Refer to Section 2.8.1 "AmbaKAL_EventFlagCreate" for flag description.
UINT32	**Option**	Specify whether all or any of the requested event flags are required 0x00: **AMBA_KAL_OR** 0x01: **AMBA_KAL_OR_CLEAR** 0x02: **AMBA_KAL_AND** 0x03: **AMBA_KAL_AND_CLEAR** Selecting **AMBA_KAL_AND** or **AMBA_KAL_AND_CLEAR** specifies that all event flags must be present in the group. Selecting **AMBA_KAL_OR** or **AMBA_KAL_OR_CLEAR** specifies that any event flag is satisfactory. Event flags that satisfy the request are cleared (set to zero) if **AMBA_KAL_AND_CLEAR** or **AMBA_KAL_OR_CLEAR** are specified.
UINT32	***pActualFlags**	Actual event flags
UINT32	**Timeout**	Timeout value in milliseconds (ms). Valid range is 0x00000000 to 0xFFFFFFFF.

*Table 2-56. Parameters for Kernel AmbaKAL API **AmbaKAL_EventFlagTake()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-57. Returns for Kernel AmbaKAL API **AmbaKAL_EventFlagTake()**.*

Example:

None

See Also:

AmbaKAL_EventFlagCreate()

For PROTReview only
Confidential

2.9.4 AmbaKAL_EventFlagGive

API Syntax:

AmbaKAL_EventFlagGive (AMBA_KAL_EVENT_FLAG_t *pEventFlag, UINT32 Flags)

Function Description:

- This function is used to give event flags to an event flag group.

Parameters:

Type	Parameter	Description
AMBA_KAL_EVENT_FLAG_t	***pEventFlag**	Pointer to an Event Flag Group Control Block
UINT32	**Flags**	Specify the event flags to give. Refer to Section 2.8.1 "AmbaKAL_EventFlagCreate" for flag description.

*Table 2-58. Parameters for Kernel AmbaKAL API **AmbaKAL_EventFlagGive()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-59. Returns for Kernel AmbaKAL API **AmbaKAL_EventFlagGive()**.*

Example:

None

See Also:

AmbaKAL_EventFlagCreate()

Confidential
For PROTRULY Only

2.9.5 AmbaKAL_EventFlagClear

API Syntax:

AmbaKAL_EventFlagClear (AMBA_KAL_EVENT_FLAG_t *pEventFlag, UINT32 Flags)

Function Description:

- This function is used to clear event flags from an event flag group.

Parameters:

Type	Parameter	Description
AMBA_KAL_EVENT_FLAG_t	***pEventFlag**	Pointer to an Event Flag Group Control Block
UINT32	**Flags**	Specify the event flags to clear. Refer to Section 2.8.1 "AmbaKAL_EventFlagCreate" for flag description.

*Table 2-60. Parameters for Kernel AmbaKAL API **AmbaKAL_EventFlagClear()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-61. Returns for Kernel AmbaKAL API **AmbaKAL_EventFlagClear()**.*

Example:

None

See Also:

AmbaKAL_EventFlagCreate()

For PROTRULY Only Confidential

2.9.6 AmbaKAL_EventFlagQuery

API Syntax:

AmbaKAL_EventFlagQuery (AMBA_KAL_EVENT_FLAG_t *pEventFlag, UINT32 *pCurFlags)

Function Description:

- This function is used to retrieve the information about the specified Event Flag.

Parameters:

Type	Parameter	Description
AMBA_KAL_EVENT_FLAG_t	***pEventFlag**	Pointer to an Event Flag Group Control Block
UINT32	***pCurFlags**	Pointer to the current Flags of the Event Flags Group

*Table 2-62. Parameters for Kernel AmbaKAL API **AmbaKAL_EventFlagQuery()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-63. Returns for Kernel AmbaKAL API **AmbaKAL_EventFlagQuery()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.10 KAL: Message Queues

This section explains the message queue, which is an object used for synchronization and communication by sending or receiving a variable-sized message. A message queue that holds a single message is commonly called a mailbox.

Confidential
For PROTRULY Only

2.10.1 AmbaKAL_MsgQueueCreate

API Syntax:

AmbaKAL_MsgQueueCreate (AMBA_KAL_MSG_QUEUE_t *pMsgQueue, void *pMsgQueueBase, UINT32 MsgSize, UINT32 MaxNumMsg)

Function Description:

- This function is used to generate a message queue with a defined configuration

Parameters:

Type	Parameter	Description
AMBA_KAL_MSG_QUEUE_t	***pMsgQueue**	Pointer to a Message Queue Control Block
void	***pMsgQueueBase**	Pointer to the message queue
UINT32	**MsgSize**	Message size in Bytes Specifies the size of each message in the queue. Message sizes range from one (1) 32-bit word to sixteen (16) 32-bit words. Valid message size options are numerical values from 1 through 16, inclusive.
UINT32	**MaxNumMsg**	Maximum number of messages. Valid range is 0x00000000 to 0xFFFFFFFF.

*Table 2-64. Parameters for Kernel AmbaKAL API **AmbaKAL_MsgQueueCreate()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-65. Returns for Kernel AmbaKAL API **AmbaKAL_MsgQueueCreate()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.10.2 AmbaKAL_MsgQueueDelete

API Syntax:

AmbaKAL_MsgQueueDelete (AMBA_KAL_MSG_QUEUE_t *pMsgQueue)

Function Description:

- This function is used to delete a specified message queue.

Parameters:

Type	Parameter	Description
AMBA_KAL_MSG_QUEUE_t	***pMsgQueue**	Pointer to a Message Queue Control Block

*Table 2-66. Parameters for Kernel AmbaKA API **AmbaKAL_MsgQueueDelete()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-67. Returns for Kernel AmbaKAL API **AmbaKAL_MsgQueueDelete()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.10.3 AmbaKAL_MsgQueueFlush

API Syntax:

AmbaKAL_MsgQueueFlush (AMBA_KAL_MSG_QUEUE_t *pMsgQueue)

Function Description:

- This function is used to flush a message queue.

Parameters:

Type	Parameter	Description
AMBA_KAL_MSG_QUEUE_t	***pMsgQueue**	Pointer to a Message Queue Control Block

*Table 2-68. Parameters for Kernel AmbaKAL API **AmbaKAL_MsgQueueFlush)**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-69. Returns for Kernel AmbaKAL API **AmbaKAL_MsgQueueFlush()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.10.4 AmbaKAL_MsgQueueReceive

API Syntax:

AmbaKAL_MsgQueueReceive (AMBA_KAL_MSG_QUEUE_t *pMsgQueue, void *pMsgDest, UINT32 Timeout)

Function Description:

- This function is used to receive a message from a message queue.

Parameters:

Type	Parameter	Description
AMBA_KAL_MSG_QUEUE_t	***pMsgQueue**	Pointer to a Message Queue Control Block
Void	***pMsgDest**	Location to copy the message
UINT32	**Timeout**	Timeout value in milliseconds (ms). Valid range is 0x00000000 to 0xFFFFFFFF.

*Table 2-70. Parameters for Kernel AmbaKAL API **AmbaKAL_MsgQueueReceive()**.*

Returns:

Return	Description
0	Success
- 1	Failure

*Table 2-71. Returns for Kernel AmbaKAL API **AmbaKAL_MsgQueueReceive()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.10.5 AmbaKAL_MsgQueueSend

API Syntax:

AmbaKAL_MsgQueueSend (AMBA_KAL_MSG_QUEUE_t *pMsgQueue, void *pMsgSource, UINT32 Timeout)

Function Description:

- This function is used to send a message from a message queue.

Parameters:

Type	Parameter	Description
AMBA_KAL_MSG_QUEUE_t	***pMsgQueue**	Pointer to a Message Queue Control Block
void	***pMsgSource**	Pointer to the message
UINT32	**Timeout**	Timeout value in milliseconds (ms). Valid range is 0x00000000 to 0xFFFFFFFF.

*Table 2-72. Parameters for Kernel AmbaKAL API **AmbaKAL_MsgQueueSend()**.*

Returns:

Return	Description
0	Success
- 1	Failure

*Table 2-73. Returns for Kernel AmbaKAL API **AmbaKAL_MsgQueueSend()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.11 KAL: Application Timers

This section explains the application timers, which provide applications with the ability to execute application C functions at specific intervals of time. The actual time between timer ticks is 1ms in our implementation.

Confidential
For PROTRULY Only

2.11.1 AmbaKAL_TimerCreate

API Syntax:

AmbaKAL_TimerCreate (AMBA_KAL_TIMER_t *pTimer, UINT32 AutoStart, void (*ExpirationFunction) (UINT32), UINT32 ExpirationArg, UINT32 InitTicks, UINT32 ReloadTicks)

Function Description:

- This function is used to generate an application timer with a defined configuration

Parameters:

Type	Parameter	Description
AMBA_KAL_ TIMER_t	***pTimer**	Pointer to the Timer Control Block
UINT32	**AutoStart**	Automatic start up after creation: 0x0: **AMBA_KAL_DO_NOT_START** 0x1: **AMBA_KAL_AUTO_START**
void	**(*ExpirationFunction)**	Application function to call when the timer expires
UINT32	**ExpirationArg**	Argument for the expiration function. Valid range is 0x00000000 to 0xFFFFFFFF.
UINT32	**InitTicks**	Initial ticks for the timer expiration. Valid range is 0x00000000 to 0xFFFFFFFF.Z
UINT32	**ReloadTicks**	Reload ticks for the timer expiration after the 1st. Valid range is 0x00000000 to 0xFFFFFFFF. A value of zero yields a one-shot timer.

*Table 2-74. Parameters for Kernel AmbaKAL API **AmbaKAL_TimerCreate()**.*

Returns:

Return	Description
0	Success
- 1	Failure

*Table 2-75. Returns for Kernel AmbaKAL API **AmbaKAL_TimerCreate()**.*

Example:

None

See Also:

None

For PROTOUL Y Only Confidential

2.11.2 AmbaKAL_TimerDelete

API Syntax:

AmbaKAL_TimerDelete (AMBA_KAL_TIMER_t *pTimer)

Function Description:

- This function is used to delete a specified application timer.

Parameters:

Type	Parameter	Description
AMBA_KAL_TIMER_t	***pTimer**	Pointer to the Timer Control Block

*Table 2-76. Parameters for Kernel AmbaKAL API **AmbaKAL_TimerDelete()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-77. Returns for Kernel AmbaKAL API **AmbaKAL_TimerDelete()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.11.3 AmbaKAL_TimerStart

API Syntax:

AmbaKAL_TimerStart (AMBA_KAL_TIMER_t *pTimer)

Function Description:

- This function is used to start an application timer.

Parameters:

Type	Parameter	Description
AMBA_KAL_TIMER_t	***pTimer**	Pointer to the Timer Control Block

*Table 2-78. Parameters for Kernel AmbaKAL API **AmbaKAL_TimerStart()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-79. Returns for Kernel AmbaKAL API **AmbaKAL_TimerStart()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.11.4 AmbaKAL_TimerStop

API Syntax:

AmbaKAL_TimerStop (AMBA_KAL_TIMER_t *pTimer)

Function Description:

- This function is used to stop an application timer.

Parameters:

Type	Parameter	Description
AMBA_KAL_TIMER_t	***pTimer**	Pointer to the Timer Control Block

*Table 2-80. Parameters for Kernel AmbaKAL API **AmbaKAL_TimerStop()**.*

Returns:

Return	Description
0	Success
- 1	Failure

*Table 2-81. Returns for Kernel AmbaKAL API **AmbaKAL_TimerStop()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.11.5 AmbaKAL_TimerChange

API Syntax:

AmbaKAL_TimerChange (AMBA_KAL_TIMER_t *pTimer, UINT32 InitTicks, UINT32 ReloadTicks, UINT32 AutoStart)

Function Description:

- This function is used to modify an application timer.

Parameters:

Type	Parameter	Description
AMBA_KAL_TIMER_t	***pTimer**	Pointer to the Timer Control Block
UINT32	**InitTicks**	Initial ticks for the timer expiration. Valid range is 0x00000000 to 0xFFFFFFFF.
UINT32	**ReloadTicks**	Reload ticks for the timer expiration after the first. Valid range is 0x00000000 to 0xFFFFFFFF. A value of zero yields a one-shot timer.
UINT32	**AutoStart**	Automatic start up after creation: 0x0: **AMBA_KAL_DO_NOT_START** 0x1: **AMBA_KAL_AUTO_START**

*Table 2-82. Parameters for Kernel AmbaKAL API **AmbaKAL_TimerChange()**.*

Returns:

Return	Description
0	Success
- 1	Failure

*Table 2-83. Returns for Kernel AmbaKAL API **AmbaKAL_TimerChange()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.11.6 AmbaKAL_GetTickCount

API Syntax:

AmbaKAL_GetTickCount (void)

Function Description:

- This function is used to get the current system tick counter value.

Parameters:

None

Returns:

Return	Description
UINT32	Current tick counter value

*Table 2-84. Returns for Kernel AmbaKAL API **AmbaKAL_GetTickCount()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.12 KAL: Memory Byte Pools

This section explains the Memory byte pools, which are similar to a standard C heap, but it is possible to have multiple memory byte pools. Allocations from memory byte pools are similar to traditional malloc calls.

Confidential
For PROTRULY Only

2.12.1 AmbaKAL_MemAllocate

API Syntax:

AmbaKAL_MemAllocate (AMBA_KAL_BYTE_POOL_t *pBytePool, AMBA_MEM_CTRL_s *pMemCtrl, UINT32 MemByteSize, UINT32 Alignment)

Function Description:

- This function is used to allocate memory from a Memory Byte Pool.

Parameters:

Type	Parameter	Description
AMBA_KAL_BYTE_POOL_t	***pBytePool**	Pointer to the memory byte pool Control Block
AMBA_MEM_CTRL_s	***pMemCtrl**	Pointer to the memory control block
UINT32	**MemByteSize**	Memory size in Bytes
UINT32	**Alignment**	Data alignment in Bytes

*Table 2-85. Parameters for Kernel AmbaKAL API **AmbaKAL_MemAllocate()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-86. Returns for Kernel AmbaKAL API **AmbaKAL_MemAllocate()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.12.2 AmbaKAL_MemFree

API Syntax:

AmbaKAL_MemFree (AMBA_MEM_CTRL_s *pMemCtrl)

Function Description:

- This function is used to free the allocated memory in a Memory Byte Pool.

Parameters:

Type	Parameter	Description
AMBA_MEM_CTRL_s	***pMemCtrl**	Pointer to the memory control block

*Table 2-87. Parameters for Kernel AmbaKAL API **AmbaKAL_MemFree()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-88. Returns for Kernel AmbaKAL API **AmbaKAL_MemFree()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.12.3 AmbaKAL_BytePoolCreate

API Syntax:

AmbaKAL_BytePoolCreate (AMBA_KAL_BYTE_POOL_t *pBytePool, void *pPoolBase, UINT32 PoolByteSize)

Function Description:

- This function is used to generate a memory byte pool with a defined configuration

Parameters:

Type	Parameter	Description
AMBA_KAL_BYTE_POOL_t	***pBytePool**	Pointer to a memory Byte Pool Control Block
void	***pPoolBase**	Pointer to a memory pool
UINT32	**PoolByteSize**	Total number of bytes of the memory pool

*Table 2-89. Parameters for Kernel AmbaKAL API **AmbaKAL_BytePoolCreate()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-90. Returns for Kernel AmbaKAL API **AmbaKAL_BytePoolCreate()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.12.4 AmbaKAL_BytePoolDelete

API Syntax:

AmbaKAL_BytePoolDelete (AMBA_KAL_BYTE_POOL_t *pBytePool)

Function Description:

- This function is used to delete a specified memory byte pool.

Parameters:

Type	Parameter	Description
AMBA_KAL_BYTE_POOL_t	***pBytePool**	Pointer to a memory Byte Pool Control Block

*Table 2-91. Parameters for Kernel AmbaKAL API **AmbaKAL_BytePoolDelete()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-92. Returns for Kernel AmbaKAL API **AmbaKAL_BytePoolDelete()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.12.5 AmbaKAL_BytePoolAllocate

API Syntax:

AmbaKAL_BytePoolAllocate (AMBA_KAL_BYTE_POOL_t *pBytePool, void **pMemBase, UINT32 Mem-ByteSize, UINT32 Timeout)

Function Description:

- This function is used to allocate a defined number of bytes from a memory byte pool.

Parameters:

Type	Parameter	Description
AMBA_KAL_ BYTE_POOL_t	***pBytePool**	Pointer to a memory Byte Pool Control Block
void	****pMemBase**	Pointer to the memory base address
UINT32	**MemByteSize**	Memory size in bytes. Valid range is 0x00000000 to 0xFFFFFFFF.
UINT32	**Timeout**	Timeout value in milliseconds (ms). Valid range is 0x00000000 to 0xFFFFFFFF.

*Table 2-93. Parameters for Kernel AmbaKAL API **AmbaKAL_BytePoolAllocate()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-94. Returns for Kernel AmbaKAL API **AmbaKAL_BytePoolAllocate()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.12.6 AmbaKAL_BytePoolFree

API Syntax:

AmbaKAL_BytePoolFree (void *pMemBase)

Function Description:

- This function is used to free a defined memory area of a memory byte pool.

Parameters:

Type	Parameter	Description
void	***pMemBase**	Pointer to the memory base address

*Table 2-95. Parameters for Kernel AmbaKAL API **AmbaKAL_BytePoolFree()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-96. Returns for Kernel AmbaKAL API **AmbaKAL_BytePoolFree()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.12.7 AmbaKAL_BytePoolInfoGet

API Syntax:

AmbaKAL_BytePoolInfoGet (AMBA_KAL_BYTE_POOL_t *pBytePool, UINT32 *pPoolFreeByteSize)

Function Description:

- This function is used to get the information of a memory byte pool.

Parameters:

Type	Parameter	Description
AMBA_KAL_BYTE_POOL_t	***pBytePool**	Pointer to the memory base address
UINT32	***pPoolFreeByteSize**	Pointer to storage of free byte size of the memory pool

*Table 2-97. Parameters for Kernel AmbaKAL API **AmbaKAL_BytePoolInfoGet()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-98. Returns for Kernel AmbaKAL API **AmbaKAL_BytePoolInfoGet()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.13 KAL: Memory Block Pools

This section explains the memory block pools, which provide the ability to create and manage multiple pools of fixed-size memory blocks.

Confidential
For PROTRULY Only

2.13.1 AmbaKAL_BlockPoolCreate

API Syntax:

AmbaKAL_BlockPoolCreate (AMBA_KAL_BLOCK_POOL_t *pBlockPool, UINT32 BlockByteSize, void *pPoolBase, UINT32 PoolByteSize)

Function Description:

- This function is used to generate a memory block pool with a defined configuration

Parameters:

Type	Parameter	Description
AMBA_KAL_ BLOCK_POOL_t	***pBlockPool**	Pointer to a memory Block Pool Control Block
UINT32	**BlockByteSize**	Block size in bytes. Valid range is 0x00000000 to 0xFFFFFFFF.
void	***pPoolBase**	Pointer to a memory pool
UINT32	**PoolByteSize**	Total number of bytes of the memory block pool. Valid range is 0x00000000 to 0xFFFFFFFF.

*Table 2-99. Parameters for Kernel AmbaKAL API **AmbaKAL_BlockPoolCreate()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-100. Returns for Kernel AmbaKAL API **AmbaKAL_BlockPoolCreate()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.13.2 AmbaKAL_BlockPoolDelete

API Syntax:

AmbaKAL_BlockPoolDelete (AMBA_KAL_BLOCK_POOL_t *pBlockPool)

Function Description:

- This function is used to delete a specified memory block pool.

Parameters:

Type	Parameter	Description
AMBA_KAL_BLOCK_POOL_t	***pBlockPool**	Pointer to a memory Block Pool Control Block

*Table 2-101. Parameters for Kernel AmbaKAL API **AmbaKAL_BlockPoolDelete()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-102. Returns for Kernel AmbaKAL API **AmbaKAL_BlockPoolDelete()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.13.3 AmbaKAL_BlockPoolAllocate

API Syntax:

AmbaKAL_BlockPoolAllocate (AMBA_KAL_BLOCK_POOL_t *pBlockPool, void **pBlockBase, UINT32 Timeout)

Function Description:

- This function is used to allocate a defined number of bytes from a memory block pool.

Parameters:

Type	Parameter	Description
AMBA_KAL_BLOCK_POOL_t	***pBlockPool**	Pointer to a memory Block Pool Control Block
void	****pBlockBase**	Pointer to the memory block start address
UINT32	**Timeout**	Timeout value in milliseconds (ms). Valid range is 0x00000000 to 0xFFFFFFFF.

*Table 2-103. Parameters for Kernel AmbaKAL API **AmbaKAL_BlockPoolAllocate()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-104. Returns for Kernel AmbaKAL API **AmbaKAL_BlockPoolAllocate()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

2.13.4 AmbaKAL_BlockPoolFree

API Syntax:

AmbaKAL_BlockPoolFree (void *pBlockBase)

Function Description:

- This function is used to free a defined memory area of a memory block pool.

Parameters:

Type	Parameter	Description
void	***pBlockBase**	Pointer to the memory block start address

*Table 2-105. Parameters for Kernel AmbaKAL API **AmbaKAL_BlockPoolFree()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-106. Returns for Kernel AmbaKAL API **AmbaKAL_BlockPoolFree()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.14 KAL: Symmetric Multi-Processing (SMP) support

This section explains the ability to control how many multi-processor cores used in SMP system.

Confidential
For PROTRULY Only

2.14.1 AmbaKAL_TaskSmpCoreExclusionSet

API Syntax:

AmbaKAL_TaskSmpCoreExclusionSet (AMBA_KAL_TASK_t *pTask, UINT32 ExclusionMap)

Function Description:

- This function is used to assign a CPU core exclusion map to a software task.

Parameters:

Type	Parameter	Description
AMBA_KAL_TASK_t	***pTask**	Pointer to the Software Task Control Block
UINT32	**ExclusionMap**	Bit map with set bits that indicate a core is excluded. Supplying a 0 value enables the thread to execute on any core (default).

*Table 2-107. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskSmpCoreExclusionSet()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-108. Returns for Kernel AmbaKAL API **AmbaKAL_TaskSmpCoreExclusionSet()**.*

Example:

None

See Also:

None

Confidential
For PROFRULY Only

2.14.2 AmbaKAL_TaskSmpCoreExclusionGet

API Syntax:

AmbaKAL_TaskSmpCoreExclusionGet (AMBA_KAL_TASK_t *pTask, UINT32 *pExclusionMap)

Function Description:

- This function is used to retrieve the CPU core exclusion map assigned to a software task.

Parameters:

Type	Parameter	Description
AMBA_KAL_TASK_t	***pTask**	Pointer to the Software Task Control Block
UINT32	***pExclusionMap**	Pointer to the Core exclusion map

*Table 2-109. Parameters for Kernel AmbaKAL API **AmbaKAL_TaskSmpCoreExclusionGet()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-110. Returns for Kernel AmbaKAL API **AmbaKAL_TaskSmpCoreExclusionGet()**.*

Example:

None

See Also:

None

Confidential
For PROTRULY Only

2.14.3 AmbaKAL_TaskSmpCurCoreGet

API Syntax:

AmbaKAL_TaskSmpCurCoreGet (void)

Function Description:

- This function is used to retrieve the current Core ID of the caller.

Parameters:

None

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-111. Returns for Kernel AmbaKAL API **AmbaKAL_TaskSmpCurCoreGet()**.*

Example:

None

See Also:

None

For Confidential PROTRULY Only

2.14.4 AmbaKAL_TimerSmpCoreExclusionSet

API Syntax:

AmbaKAL_TimerSmpCoreExclusionSet (AMBA_KAL_TIMER_t *pTimer, UINT32 ExclusionMap)

Function Description:

- This function is used to assign a CPU core exclusion map to a timer.

Parameters:

Type	Parameter	Description
AMBA_KAL_TIMER_t	***pTimer**	Pointer to the Timer Control Block
UINT32	**ExclusionMap**	Bit map with set bits that indicate a core is excluded. Supplying a 0 value enables the thread to execute on any core (default).

*Table 2-112. Parameters for Kernel AmbaKAL API **AmbaKAL_TimerSmpCoreExclusionSet()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-113. Returns for Kernel AmbaKAL API **AmbaKAL_TimerSmpCoreExclusionSet()**.*

Example:

None

See Also:

None

Confidential
For PROFRULY Only

2.14.5 AmbaKAL_TimerSmpCoreExclusionGet

API Syntax:

AmbaKAL_TimerSmpCoreExclusionGet (AMBA_KAL_TIMER_t *pTimer, UINT32 *pExclusionMap)

Function Description:

- This function is used to retrieve the CPU core exclusion map assigned to a timer.

Parameters:

Type	Parameter	Description
AMBA_KAL_TIMER_t	***pTimer**	Pointer to the Timer Control Block
UINT32	***pExclusionMap**	Pointer to the Core exclusion map

*Table 2-114. Parameters for Kernel AmbaKAL API **AmbaKAL_TimerSmpCoreExclusionGet()**.*

Returns:

Return	Description
0	Success
-1	Failure

*Table 2-115. Returns for Kernel AmbaKAL API **AmbaKAL_TimerSmpCoreExclusionGet()**.*

Example:

None

See Also:

None

For PROTRULY Only Confidential

Appendix 1 Additional Resources

Related resources include:

- *AMBARELLA SDK6 API: Kernel Abstraction Layer (AmbaKAL)*
- *AMBARELLA SDK6 API: File System (AmbaFS)*
- *AMBARELLA SDK6 API: System (AmbaSYS)*
- *AMBARELLA SDK6 AN: ADC and IR Input*
- *AMBARELLA SDK6 AN: Audio Codec Driver*
- *AMBARELLA SDK6 AN: Custom Image Sensor Driver*
- *AMBARELLA SDK6 AN: Custom LCD Panel Driver*

Please contact an Ambarella representative for a full list of related resources.

Confidential
For PROTRULY Only

Appendix 2 Important Notice

All Ambarella design specifications, datasheets, drawings, files, and other documents (together and separately, "materials") are provided on an "*as is*" basis, and Ambarella makes no warranties, expressed, implied, statutory, or otherwise with respect to the materials, and expressly disclaims all implied warranties of noninfringement, merchantability, and fitness for a particular purpose. The information contained herein is believed to be accurate and reliable. However, Ambarella assumes no responsibility for the consequences of use of such information.

Ambarella Incorporated reserves the right to correct, modify, enhance, improve, and otherwise change its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

All products are sold subject to Ambarella's terms and conditions of sale supplied at the time of order acknowledgment. Ambarella warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with its standard warranty. Testing and other quality control techniques are used to the extent Ambarella deems necessary to support this warranty.

Ambarella assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using Ambarella components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

Ambarella does not warrant or represent that any license, either expressed or implied, is granted under any Ambarella patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which Ambarella products or services are used. Information published by Ambarella regarding third-party products or services does not constitute a license from Ambarella to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from Ambarella under the patents or other intellectual property of Ambarella.

Reproduction of information from Ambarella documents is not permissible without prior approval from Ambarella.

Ambarella products are not authorized for use in safety-critical applications (such as life support) where a failure of the product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Customers acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of Ambarella products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by Ambarella. Further, Customers must fully indemnify Ambarella and its representatives against any damages arising out of the use of Ambarella products in such safety-critical applications.

Ambarella products are neither designed nor intended for use in military/aerospace applications or environments. Customers acknowledge and agree that any such use of Ambarella products is solely at the Customer's risk, and they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

Confidential
For PROTRUY Only

Appendix 3 Revision History

NOTE: Page numbers for previous drafts may differ from page numbers in the current version.

Version	Date	Comments
	26 Feb 2013	Formatting
	12 Mar 2013	Revised the title page and Appendix1
	19 Mar 2013	Add AmbaKAL_TaskReset, AmbaKAL_TaskQuery
0.1	29 Mar 2013	Updates from Taiwan
1.0	17 Apr 2013	Prepare for release
1.5	9 Oct 2013	Refine all descriptions; formatting
1.6	30 Oct 2013	Modify Chapter 2. Add one API.
1.7	2 October 2014	Formatted to SDK6, add new APIs: AmbaKAL_TaskFpuEnable AmbaKAL_TaskFpuDisable AmbaKAL_EnterCriticalSection AmbaKAL_ExitCriticalSection AmbaKAL_BytePoolInfoGet.
1.8	5 June 2015	Add Sections 2.5 KAL: User Specific, 2.5.1 AmbaKAL_TaskUserNotify, 2.5.2 AmbaKAL_TaskUserValueGet, 2.5.3 AmbaKAL_TaskUserValueSet, 2.5.4 AmbaKAL_RegisterStackErrorHandler, 2.5.5 AmbaKAL_IsInISR, 2.9.6 AmbaKAL_EventFlagQuery and 2.11.6 AmbaKAL_GetTickCount.

Table A3-1. Revision History.

For PROTRULY Only Confidential