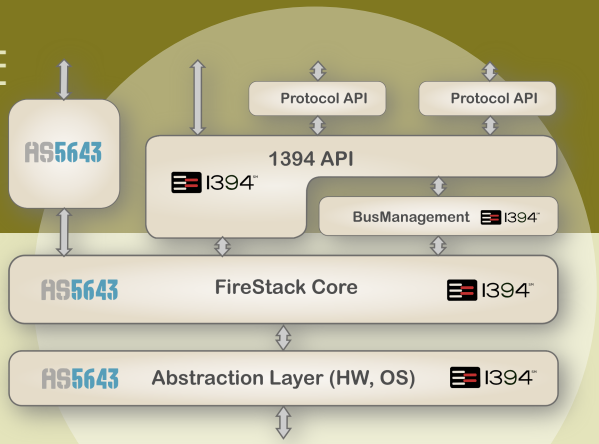
dap TECHNOLOGY

SOFTWARE

FIRESTACK



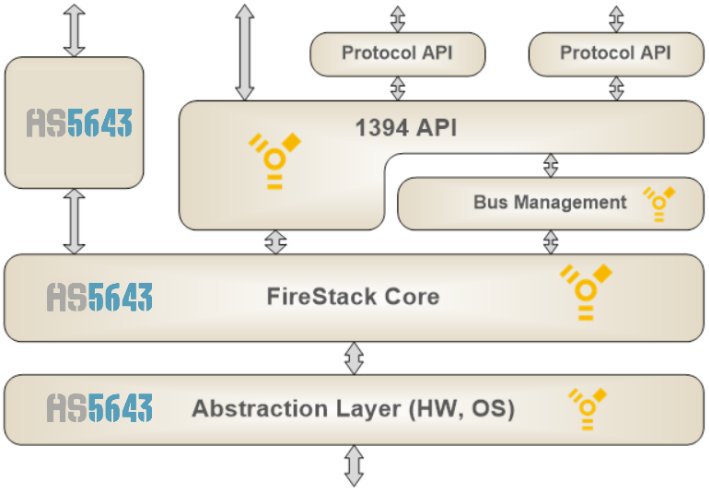
1394SM

MIL1394

Product Overview

FireStack® is DapTechnology's homegrown 1394 software stack. Initially developed in conjunction with the company's 1394 Link Layer Controller FireLink®, the product was architected from ground up in order to support the advanced features of the FireLink® IP (Basic and Extended) solution. Since then additional mechanisms have been added to FireStack® to enable support for generic OHCI compliant Link layers.

But FireStack® is also a specialty software stack as it targets usage in areas where requirements deviate a bit from the "standard" needs. For example, PCs and consumer electronics implementations differ significantly from their industrial, vision and aerospace counterparts. In many of these latter examples the focus lies on performance, latency, robustness, etc.



Conceptually, FireStack® addresses all these issues. It complies with IEEE1394 requirements as well as select higher protocol layers e.g. IIDC). And as the only product in the market it natively supports AS5643 features and functions. Due to the time critical aspects of that AS5643 protocol the AS5643 portion of FireStack® is only supported when running on top of an LLC with embedded AS5643 HW support (see FireLink® and FireTrac®).

FireStack® is also an alternative to using general purpose SW implementations with standard OHCI I/O interfaces as it offers a more deterministic approach. Modern plug & play solutions are not always recommended for use in enclosed systems since they typically add undesired bus traffic, latencies, etc. The ability to customize FireStack® modules allows focusing on the task requirements without unwanted system, traffic and data overhead.

Operating System Support

FireStack's OS support is targeted to support the major systems used in the market segments described above. At the moment OS support exists for:

Windows, LabVIEW, Linux, VxWorks, QNX, LabVIEW Real-Time, RTX64 Real-time 64-bit Windows IntervalZero

Support for other operating systems will be added as the need arises. Please contact DapTechnology for your specific requirements.

Optimized Transaction Management

FireStack® features an innovative 1394 packet handler, whose objective is to reduce resource burden when receiving and transmitting 1394 packets. The consistent utilization of zero-copy operations greatly enhances the overall system performance. FireStack® provides memory buffers accessible by both the user application and the 1394 Link Layer DMA engine.

For example, when transmitting an asynchronous packet and the user application has filled the memory buffer with the needed packet data, FireStack® will hand the buffer directly to the Link Layer for reading the packet into the packet transmission FIFO without having the CPU copy memory to memory.

Likewise, displaying a video stream from an IIDC camera only requires creation of DMA-capable reception buffers for the video frame data and registration of notification upon filling of a complete frame. Once notified the buffers holding the received data can be accessed directly by the video rendering engine in order to move the data to video card memory.

Inbound Transactions (IBT):

Inbound Transactions (handling of incoming requests) are defined in two separate methods:

Map Local Memory: The user can "map" a memory buffer to a specific address space. The contents of the memory buffer can be accessed by the user application at any point in time. At the same time when the stack receives a request packet from a remote device it will automatically perform the response operation (read, write or lock) and will send a response packet back to the requester. The user has the option to be notified by the notification callback function when the transaction completes.

Transaction Handler: Similar to the Map Local Memory above the user can "register for" a specific address space instead of "mapping" local memory. When the stack receives a request packet it will call the user-specified handler callback function. The user can then perform any operation within the callback function. Upon returning from the handler callback, the FireStack® may transmit a response packet. The notification callback function will be called after completion of the response process.

Outbound Transactions (OBT):

This module can be used to perform memory transactions (read, write, lock) on remote nodes. When a memory transaction is initiated FireStack® will automatically determine the maximum speed to the destination node by performing the needed PHY remote accesses.

Outbound Transactions can be used in the following ways with respect to result indication: In ***Blocking mode*** the TX functions will not return until the response packet is received and thus making the code sequential. In contrast to that the ***Non-Blocking mode*** can be used to initiate a series of "split" transactions i.e. a series of requests that are then followed by the corresponding - but not necessarily sequential - responses.

Isochronous Messaging:

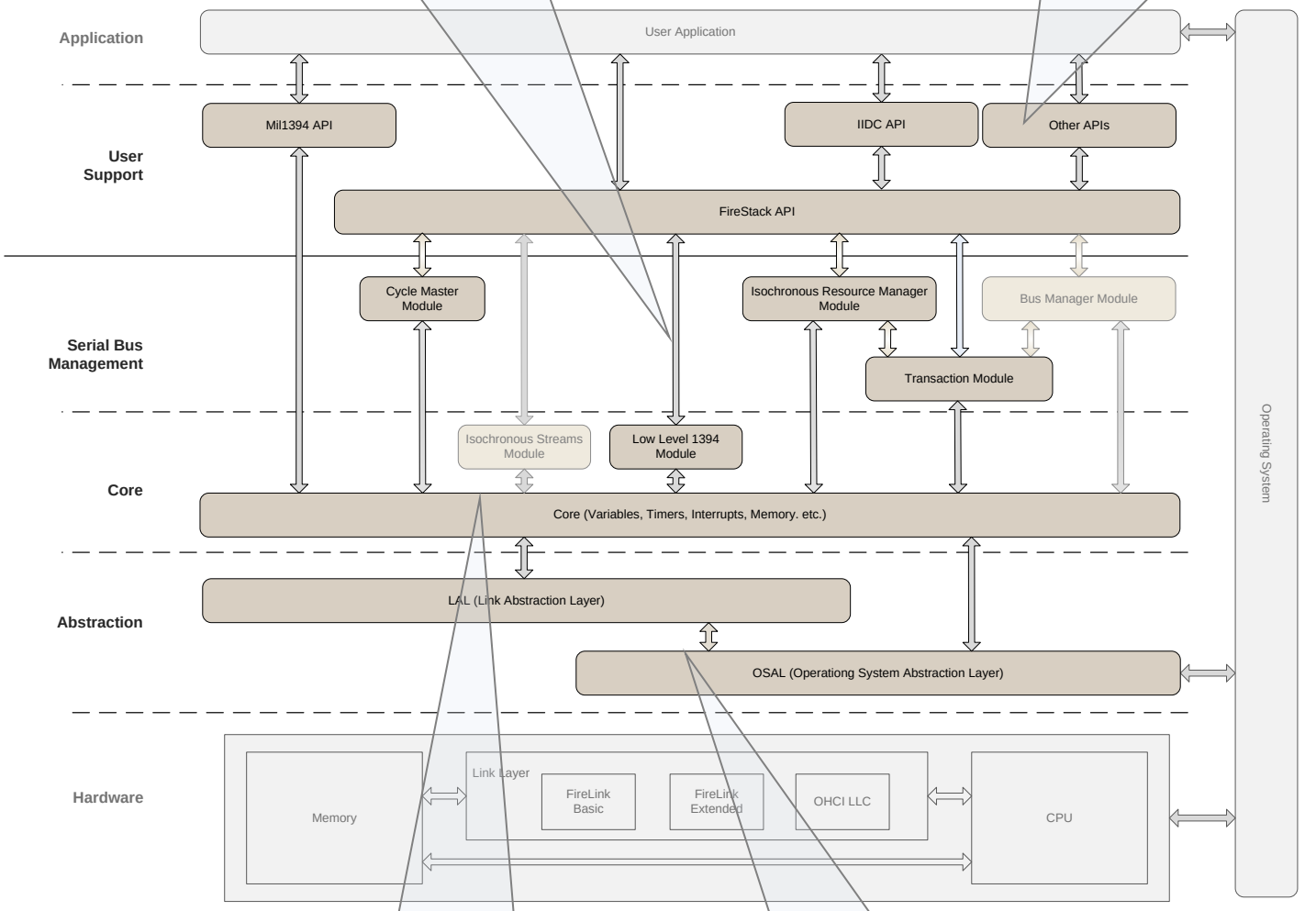
The Isochronous Streaming modules build on the mechanisms for efficient data processing defined by OHCI and provide a flexible and user-friendly API around it. Linked lists of buffers can be setup by the user application and will be automatically processed by the Link Layer DMA engine. Notification mechanisms are available for buffer and/or packet completion.

Serial Bus Management

FireStack® Serial Bus Management modules define protocols, services, and operating procedures on how nodes can govern the operation of the remaining nodes on the bus. Together with CSR, configuration ROM facilities are used to configure and manage the activities at an individual node. Bus Manager, Isochronous Resource Manager and Cycle Master are optional modules as they are not required for all 1394 implementations.

Application Programming Interface

FireStack® offers a near OS independent (user-space) APIs (C++ and Python) with complete 1394 bus control. Additionally, higher-level layers will be and are being developed on top of that to provide a very task oriented API. The APIs extend to Mil1394 and are targeting usage in the aerospace world without requiring programmers to work at the basic 1394 level.



FireStack® Core

FireStack® Core forms the engine of the software stack. Core processes all functions such as user notification, interrupt event handling, background processing tasks, packet (isochronous and asynchronous) receive and transmit, memory allocations, timers, etc.. FireStack® Core is not specific to any device or OS and is designed to run on top of the Link and OS Abstraction Layers.

Abstraction Layers

Both the Link Abstraction Layer (LAL) as well as the OS Abstraction Layer (OSAL) form the direct HW and operating system interfaces for FireStack®. They are essential for allowing FireStack® to run on a variety of HW/FW Link Layer implementations, as well as for supporting several operating systems without changing the upper layer stacks conceptually.

AS5643 Module

The SAE-AS5643 protocol differs from other 1394 protocols because of its stringent timing requirements. Because of potential inaccuracies and unpredictable latencies possible with software implementations, DapTechnology strongly believes that the AS5643 protocol timing is best implemented via a HW extension in the 1394b Link Layer. Therefore, Dap has added the AS5643 protocol timing in FireLink® Extended (FPGA IP Core block) as an add-on module. With this HW support FireLink® Extended is easily capable of meeting the AS5643 frame timing requirements and eliminates the need for complicated interrupt schemes or real-time operating systems typically needed to efficiently use the AS5643 protocol.

The FireStack® software library contains a AS5643 protocol module that can be used to control the AS5643 hardware of either a custom FireLink Extended enabled product or DapTechnology's FireTrac I/O card. This section describes how frame timing can be configured and used for both timed transmission and reception.

Frame Timing: FireStack® is very flexible in the way it handles the timing of Start of Frames. Frame synchronization for AS5643 reception and transmission may be configured as either “Free Running” or internal clock (based on a 1 microsecond input signal), based on STOF packets on the bus (just any packet on a configurable channel) or on an External Sync Input Signal.

Reception: AS5643 reception provides a filtering mechanism and all incoming packets will be run against a comprehensive verification system. Messages can be filtered on channel number, AS5643 message ID or a combination of both.

Transmission: AS5643 Transmission module can be used to control devices that support AS5643 timed transmission in hardware – as done with DapTechnology's FireTrac® and FireLink® Extended. FireTrac offers very accurate transmission timing without software intervention enabling this functionality without the need for a Real-Time operating system. The following transmission modes are available:

> ***Streaming messages:*** Allows writing large or small sets of messages to FireStack® and having them transmitted automatically at specified frame offset times. The provided data needs to contain so called frame separator elements to indicate that the following message needs to be transmitted in the next frame.
>
> ***Repeating messages:*** Allows setting up a message that will automatically be transmitted each frame by the FireStack®. The user will have a pointer to the actual data of the message and is allowed to manipulate the data at any point in time without having to worry about its timed transmission. This is very useful for AS5643 status messages.
>
> ***Single messages:*** Allows simply transmitting a message as soon as possible but exactly at the specified frame offset time. Several messages may be handed to the FireStack® for immediate transmission and the FireStack® will then take care of the actual moment of transmission.
>
> ***STOF Messages:*** Allows controlling transmission of STOF messages.

References

The FireStack® has been successfully deployed on the following DapTechnology products:

- FireTrac® AS6543 I/O card
- FireSpy® S1600 and S3200
- S1600 Host Adapter Card (optional)
- FireLink Extended evaluation kit (Xilinx PPC)

Application Programming Interface (API)

FireStack® provides APIs (C++ and Python) with multiple levels of abstraction from the 1394 bus. The ***User Support API*** allows for a very “1394-unaware” application programming. Due to the very high degree of abstraction the user does not have to be a 1394 expert for most of the typical 1394 bus control and transaction handling tasks. This high level API focuses on ease-of-use, low learning curve and streamlined programming.

For very fundamental 1394 bus controls a ***Low Level API*** is available. It can be used for operations on the basic 1394 level. For example this API allows for remote PHY access, commands, bus optimization, error condition testing, etc. As an example, it's goal is to provide CRC overwrite functionality in order to simulate erroneous bus signaling as well as other advanced and non-standard features within a SW stack. It is important to understand that some of these features are (or will be) available on specialized Link Layer controllers.

Configurability / Customization

The FireStack® architecture supports a very modular SW design. Several modules of the stack, e.g. the Isochronous Resource Manager, Cycle Master and Bus Manager, can be compiled-in or left out based on user needs. If compiled-in they can be disabled on FireStack startup (if needed).

The default FireStack® package contains all modules and license keys used to unlock individual features. Based on customer demand the deliverable can be customized with only the needed modules included in order to reduce size and increase performance.

Specifications:

- IEEE 1394-2008 compliant
  - Low-Level API (packet TX/RX, topology, ...)
  - Isochronous Streaming API
  - Inbound Transactions API
    - Memory Mapped
    - Handler Mapped
  - Outbound Transactions API
  - Serial Bus Management API
- Support for High Level Protocols
  - AS5643
  - IIDC (pending)
- Supported Link Layer Controllers:
  - FireLink® Extended and
  - FireLink® Basic (pending)
  - Select OHCI Link Layer Controllers
- Supported Operating Systems
  - Windows™
  - VxWorks (please contact for specific CPU support)
  - LabView and LabView RT
  - Linux
  - QNX
  - Xilinx PPC (please contact for porting to specific board)
  - RTX64

CONTACT INFORMATION:

sales@daptechnology.com
www.daptechnology.com

DapTechnology B.V.
Beatrixstraat 4
7573AA Oldenzaal
The Netherlands
Ph: +31 541 532941

DapUSA, Inc.
780 W San Angelo Street
Gilbert, AZ 85233
United States of America
Ph: +1 480 422 1551

DT-PRO022DAT552E, DEC2022