1394 PRINTER WORKING GROUP

IEEE 1394 HIGH SPEED BUS
DATA FIFO ADDRESS PRINTING PROFILE

*** PRELIMINARY DRAFT PROPOSAL ***

Revision 0.02- September 15, 1997

Alan Berkema
Hewlett Packard

1. OVERVIEW

The purpose of this document is to provide requirements for the implementation of IEEE 1394 communications between printers, scanners, cameras and other imaging devices.

2. IEEE 1394 ABSTRACT

IEEE 1394-1995 was ratified in December of 1995. This standard describes a high speed serial bus that has a 64 bit address space, control registers, and read/write/lock operations that conform to the ISO 13213/IEEE 1212, Command and Status Register (CSR) standard.

In addition to the standard read/write/lock transactions used for Asynchronous communications the Serial Bus provides an Isochronous data transport that guarantees latency and bandwidth.

Data transmission uses two low-voltage differential signals to connect devices at 98.304 Mbit/s, 196.608 Mbit/s, and 393.216 Mbit/s speeds.

The cable medium allows up to 16 cable hops between any two device, each hop up to 4.5 meters long, giving a total cable distance of 72 meters. Bus management recognizes smaller configurations to optimize performance. The physical topology for the cable environment is a non-cyclic network (no loops) with the only limitation being the number of cable hops between any two devices (called “nodes”) and the length of a cable hop. The cables consist of 2 shielded twisted pairs for signals and one pair for power and ground. These cables connect to “ports” on the nodes. Each port consists of terminators, transceivers and simple logic. The cable and ports act as bus repeaters between the nodes to simulate a single logical bus. The Serial Bus also uses a fair bus access mechanism that guarantees all nodes equal access.
[5 Teener].

Plug and play is supported through automatic Node Identification without the need for switches or terminators.

The Serial Bus is not a network or an I/O channel, it is a shared memory architecture. The 64 bit address is divided into 16 bits for the Node ID (node number and bus number) and 48 bits (256 terabytes) for the memory space and CSR registers.

The 1394 standardization effort continues with p1394.a and p1394.b. P1394.a is focused on correcting ambiguities and problems in 1394-1995 as well as new enhancements such as Arbitrated Reset, Fly-By-Arbitration and Ack Acceleration. P1394.b is focused on higher speeds (800 Mbit/s) as well as long distance considerations using Plastic Optical Fiber.

3. References

1. ISO/IEC 13213:1994, Control and Status Register Architecture for Microcomputer Buses
2. IEEE Std 1394-1995, Standard for High Performance Serial Bus
3. Serial Bus Protocol 2, Revision T10/1155D
4. Software Design for IEEE 1394 Peripherals, Peter Johansson.
5. New Technology in the IEEE P1394 Serial Bus, Michael Teener, March 29, 1994.

4. Control & Status Registers (CSR)

The basic functionality of a Transaction Capable 1394 node includes fundamental PHY repeater operation as well as the implementation of certain core CSR's. The CSRs are defined within the initial register space beginning at offset 0xFFFF F000 0000. This allows the node to participate in Asynchronous read/write/lock transactions. The core CSRs are specified below:

STATE_CLEAR and STATE_SET. All bits within these registers are optional but the registers themselves must be present.

NODE_IDS. Used to identify the 16-bit address of the node. In the cable environment, the LINK and PHY must together initialize this register during the self identify process that follows a bus reset.

RESET_START. This is a write-only register available to force a command reset of the node, as defined by ISO/IEC 13213:1994.

SPLIT_TIMEOUT In order to make requests, the node must implement a transaction time-out capability and make it configurable (or at least readable) through the SPLIT_TIMEOUT register.

5. Requirements For Isochronous Data Transmission

Serial Bus nodes that can participate in Isochronous operations, either as a talker or a listener, must have all of the features of transaction capable nodes. Additional requirements support the timing and detection of Isochronous operations. A key element in Serial Bus is that all Isochronous nodes share the same, coordinated time. Because Serial Bus is a distributed environment, each node must have its own 24.576 MHz cycle clock which runs freely when the node's link layer is active. This clock must be visible through the CYCLE_TIME register. Jitter in these clocks is eliminated every 125 usecs by the appearance of a cycle start packet on Serial Bus. A cycle start packet is essentially a write to the CYCLE_TIME register with a resynchronization value that eliminates jitter.

An Isochronous node must implement its resynchronization logic such that Serial Bus time, as observed by the values of the CYCLE_TIME register, can never give the appearance of moving backward. Although optional, Isochronous operations are expected to be inexpensive to implement within a node.

In addition to the requirements to talk or listen during Isochronous cycles, at least one node on a Serial Bus must be able to be the source of the synchronized cycle clock. This node is referred to as the **cycle master**. The cycle master must be able to use its 24.576 MHz clock to trigger cycle start events 8,000 times a second. As soon after a cycle start that the cycle master is able to arbitrate for the bus, it transmits a cycle start packet to resynchronize the clocks at all Isochronous nodes.

In addition to acting as the source for the Serial Bus cycle clock, a cycle master also has to implement the BUS_TIME register. This register is needed to extend the range of the Serial Bus clock from the limit permitted by the CYCLE_TIME register (about two minutes) to approximately 136 years.

An **Isochronous resource manager** is not actually an active manager of resources so much as it is an agreed upon location where all the Serial Bus nodes can cooperatively record their use of Isochronous resources, channel and bandwidth. BANDWIDTH_AVAILABLE is a register that stores the amount of Isochronous cycle time available to transmit data. Before any bandwidth is allocated, the register is initialized to a value that represents approximately 100 usecs. This permits some time to be implicitly reserved for asynchronous operations. CHANNELS_AVAILABLE is a register that is a bit map of all 64 possible Isochronous channels, free or in use. The Isochronous resource manager has another function, and that is to point to the bus manager, if any. Like the two CSR's just described, the BUS_MANAGER_ID register is a known location that is initialized by the node that assumes the role of the bus manager. It's important to note that a node that is Isochronous resource manager capable must be able to not only participate in the self identify process but to also analyze all of the self-ID packets observed. This is because if there are more than one Isochronous resource manager capable nodes on Serial Bus, only one becomes the Isochronous resource manager — the one with the largest 6-bit physical ID.
[4 Johansson]

It is recommended that all Isochronous nodes also be cycle master capable.

6. 1394 Bus Management

The highest level of functionality available to a Serial Bus node is that of the **bus manager**. Bus managers need all of the capabilities discussed in the preceding sections plus additional intelligence to analyze the Serial Bus configuration and optimize it. It is likely that bus manager capabilities are implemented in firmware. They do not require additional hardware support as much as some sort of processor to perform the analysis. This does not, however, preclude the design of a bus manager entirely in logic. The key functions of a bus manager are: If Isochronous operations are desired but the current root node is not cycle master capable, the bus manager must perform a bus reset and insure that the new root can be the cycle master. The bus manager must collect and analyze the self-ID packets in order to make the TOPOLOGY_MAP and SPEED_MAP registers available. Serial Bus management applications need to communicate this information to system administrators. Serial Bus has a configurable parameter, the gap count, which is initially set to a default value. The bus manager can significantly improve Serial Bus performance by setting the gap count to a smaller value, the minimum which can be determined from maximum hop count of the current topology. Power management permits the bus manager to intelligently enable and disable selected nodes if the aggregate power demands are greater than the power available. An important distinction to remember about the bus manager is that it is an autonomous entity that has functions to perform even if no API is provided to a hypothetical bus management application. Also keep in mind that the costs associated with implementing a bus manager are not likely to be silicon costs, but testing costs. There can be a complex matrix of test cases to cover.
[4 Johansson]

7. CSR Summary

BUSY_TIMEOUT is needed for Asynchronous retry transactions.

Core Registers

Offset	Register	Initial Value
0x000	STATE_CLEAR	
0x004	STATE_SET	
0x008	NODE_IDS	
0c00C	RESET_START	
0x018-01C	SPLIT_TIME_OUT	

Serial Bus Dependent

Cycle Master

Offset	Register	Initial Value
0x200	CYCLE_TIME	
0x204	BUS_TIME	

Other Serial Bus Dependent

Offset	Register	Initial Value
0x210	BUSY_TIMEOUT	

Isochronous Resource Manager

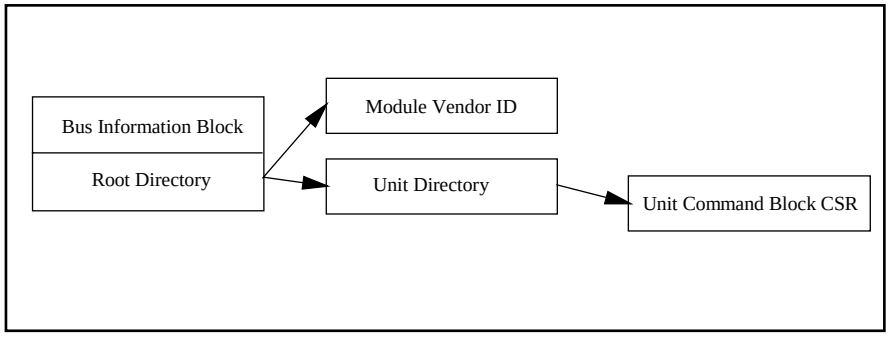
Offset	Register	Initial Value
0x21C	BUS_MANAGER_ID	
0x220	BANDWIDTH_AVAILABLE	4915
0x224-228	CHANNELS_AVAILABLE	All Ones

See the IEEE 1394-1995 specification for detailed information about each register.

8. Configuration ROM

PWG 1394 nodes shall implement configuration ROM as defined in ISO/IEC 13213:1994, IEEE Std 1394-1995, and SBP-2. The ROM directory structure is a hierarchy of information blocks with the blocks higher in the structure pointing to the blocks beneath them. The locations of the initial blocks, *Bus_Info_Block* and *Root_Directory*, are fixed. The locations of the other entries are specified in the *Root_Directory* and its associated directories.

The block diagram below illustrates device Configuration ROM relationships. Additional directories are defined in following sections.



Note:
Length values in the Configuration ROM specify the number of Quadlets.
Offsets are Direct. Address of Offset Register plus 4 times Offset value.
Reserved fields shall be set to zero.

First Quadlet
Offset: 0x400

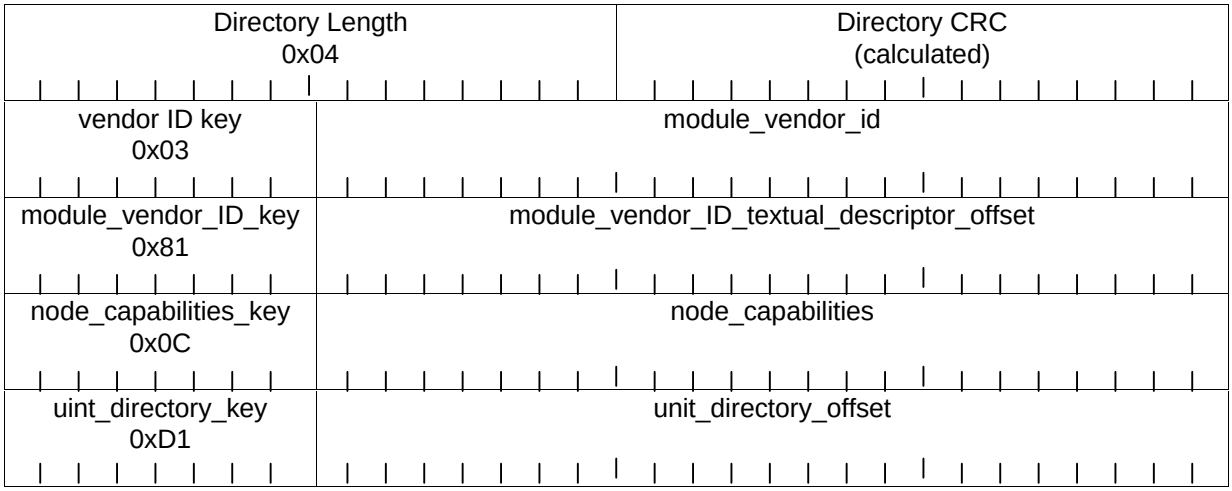
bus_info_length 0x04	CRC_length 0x04	ROM_CRC_value (calculated)

Bus Information Block
Offset: 0x404

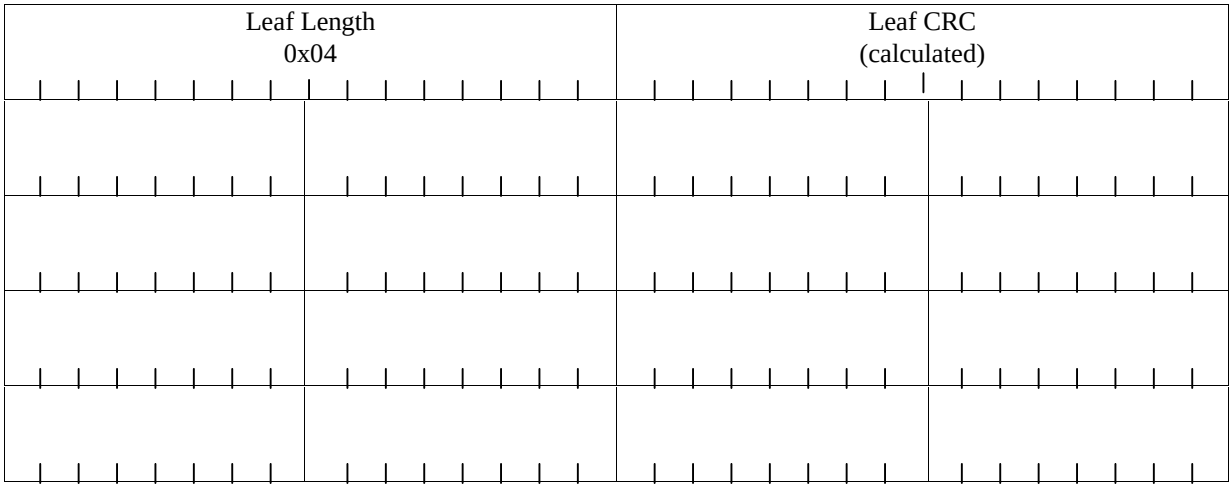
0x31					0x33	0x39	0x34
IRC	CMC	ISC	BMC	reserved	cyc_clk_acc	max_rec 0x1001	reserved 0x00
node_vendor_id (same as module_vendor_id)							chip_id_high
chip_id_low							

Taken together the node_vender_id, chip_id_high and chip_id_low are the EUI-64 also known as the Global Unique Identifier (GUID).

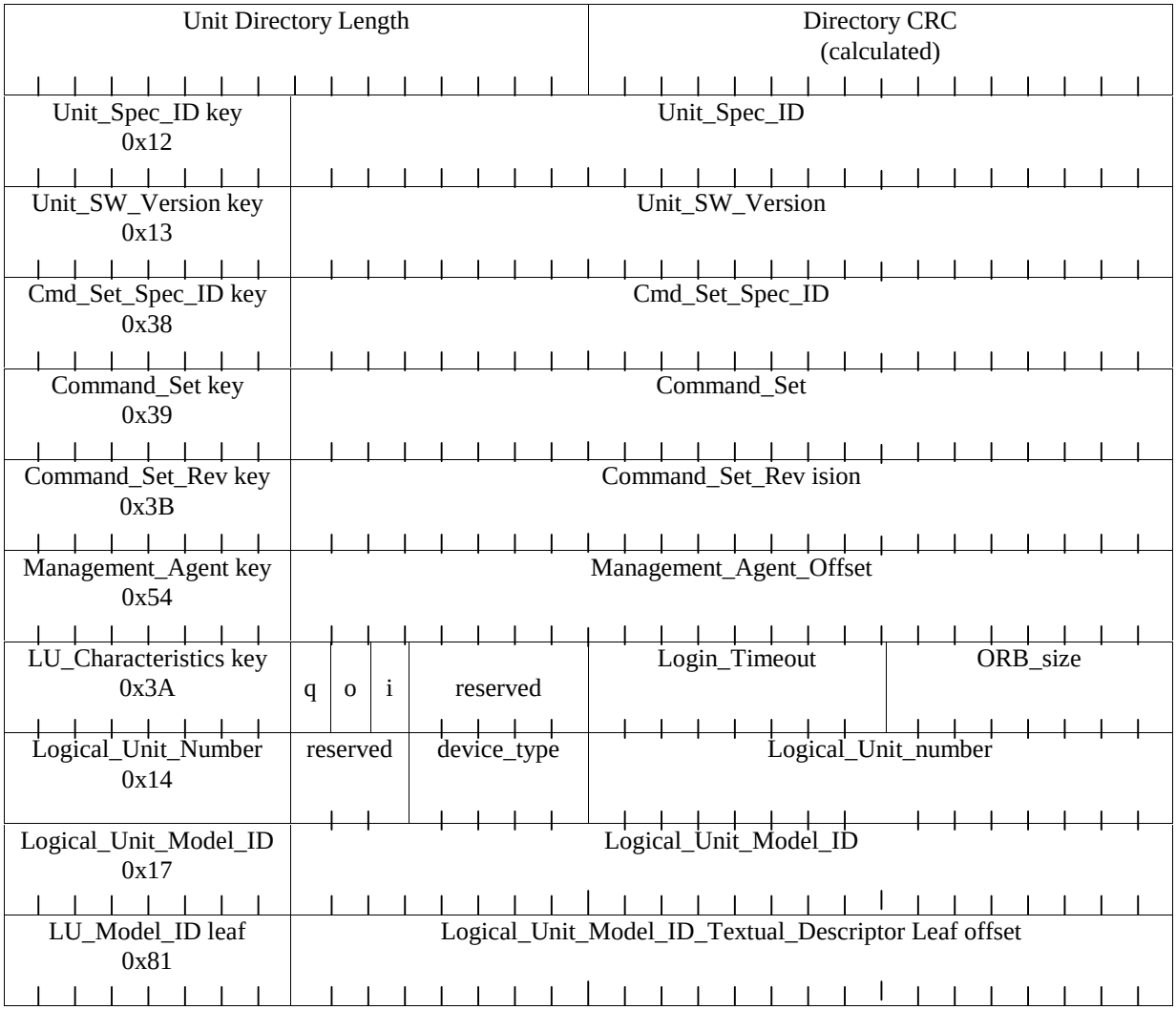
Root Directory
Offset: 0x414



Module_vendor_ID_textual_descriptor
Offset: 0x428



Unit Directory
Offset: 0x43C



Values for unspecified fields is TBD.

Unit Command_Block_Agent CSRs. This Address is returned in the Login Response.
Address: 0xFFFF F001 0000

Relative Offset	Name	Description
0x00	Agent_State	Reports fetch Agent State
0x04	Agent_Reset	Resets fetch agent
0x08	ORB_Pointer	Address of ORB
0x10	Doorbell	Signals fetch agent to refetch an address pointer
0x14	Unsolicited_Status_Enable	Acknowledges the Initiator's receipt of unsolicited status
0x18 - 0x1C		Reserved

9. 1394 PWG Peripheral Communications Requirements

- Access Control
- Fair Access
- Determine How Many Logins
- In Order Data Delivery
- Flow Control
- Guaranteed Delivery
- Error Detection
- Correction/Recovery
- Multiple Independent Channels
- Single Channel is bi-directional
- PDL, Application, OS Independent
- Standard Will Allow Concurrent Operation of Multiple Protocols
- True End of Job Detection

10. Protocol Proposal Comparison

The 1394 PWG has explored several options for peripheral communications protocols. In general proposals have gravitated towards SBP-2 and IEC 61883 (FCP). As these were examined with respect to the list of 1394 PWG requirements both of these seem to fall short.

SBP-2

- Even though the direction bit in a command ORB allows reads or writes, communication is not truly bi-directional. The direction bit works well if the Initiator can predict in advance the exact amount of data that will be communicated in both directions. Mass storage is a good example. The Initiator knows exactly how many disk blocks will be read or written and it is possible to build a linked list of Command ORBs, which accomplish these reads, and writes. With peripheral communication, commands are often embedded in the data stream (Postscript or PCL). The number of bytes that flow in either direction could be dynamic. It is difficult for the transport layer, which should be application independent, to know how much information will flow in either direction.

- Attempts to work around these SBP-2 shortcomings, such as multiple fetch agents or dual Logins, seem excessively complex especially when communication between several Initiators and Targets are considered.

- For Consumer devices the whole idea of ORB fetching is considered a “Heavy Protocol”.

FCP

- FCP seems to be optimized for a point to point connection only.

- It does not provide a mechanism for controlling which devices may access the command or response registers.

- The command and response registers are at fixed locations in the configuration ROM space.

- For the reasons above FCP, cannot be easily extended for communications between multiple devices.

Data_FIFO_Address Method (DFA)

- The DFA method is borrowed from IP1394 and is similar to the basic operation of FCP.

- Provides for true bi-directional communication.

- Efficient use of 1394 unified block write transactions.

- Low complexity allows it to be easily explained and specified

- Command set independent.

- Allows for the addition of higher layer protocols.

- Logins allow for Access control and management functions.
- Query Logins allows for limited “Loginless” status

ALL Protocols

- Do not provide explicit Flow Control.
- Do not allow for Multiple Logical Channels.
- Do specify Fairness.

11. DFA Protocol Overview

This profile divides 1394 communications into three categories

1. Management Functions
2. Asynchronous Data Transfer
3. Isochronous Data Transfer

Management functions, such as Login, Unsolicited Status, and Reconnect, are used to establish and manage 1394 connections and communication paths.

The terms Initiator and Target have a specific meaning derived from SBP-2 and do not imply the direction of data transfer.

Initiator: Originates management functions such as Login and Reconnect
Target: Responds to management functions and generates Unsolicited status.

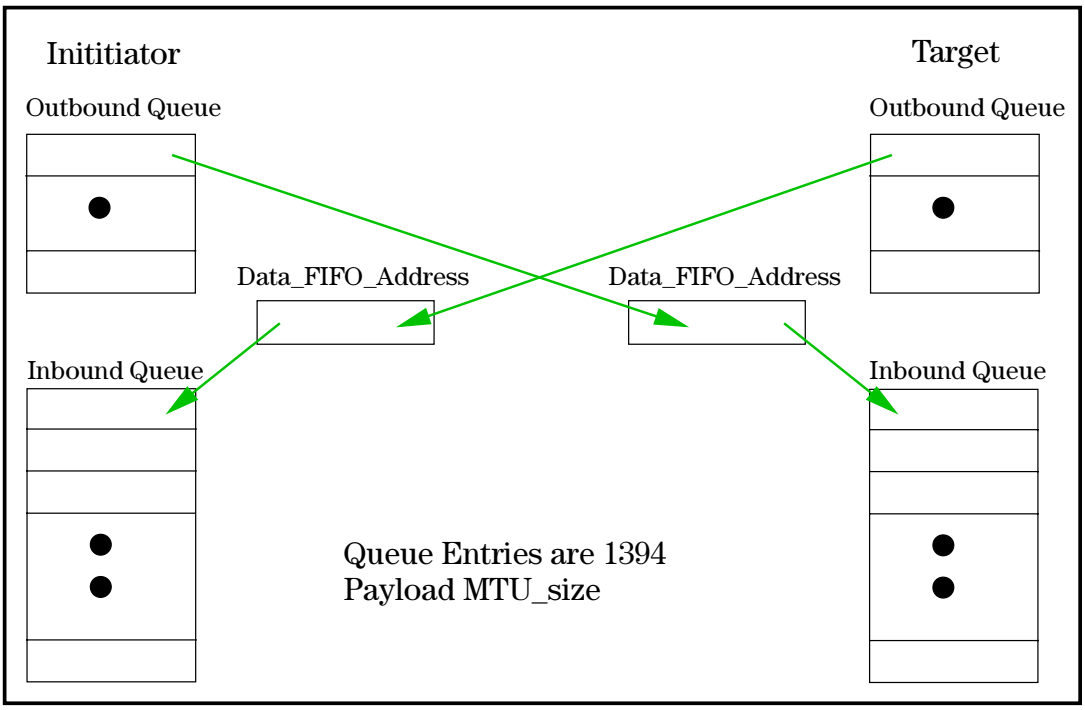
1. Discovery is TBD.
2. Login & Login Response - Will use the SBP-2-like Login and Response approach.
3. Data_FIFO_Address Exchange - occurs during login
4. Asynchronous Data Transfer will be accomplished using the Data_FIFO_Address method.
5. Isochronous Data Transfer is TBD.
6. Query Logins will use the SBP-2 mechanism.
7. Unsolicited Status will use the SBP-2 mechanism.
8. Reconnect (after 1394 Bus Reset) will use the SBP-2 mechanism.
9. Logout is TBD.

12. Discovery

Discovery is currently being addressed by the 1394 PWG.
Devices perform discovery when necessary after power on, warm boot, cold boot and 1394 bus reset.

13. Asynchronous Transmission and the Data_FIFO_Address Method

This approach is similar to what IP1394 has proposed. In IP1394 each device has a special address that both sides write to. For large (multiple packets) data transfer this scheme requires flow control at a higher layer.



The Initiator and Target each possess a Data_FIFO_Address as illustrated in the diagram above. All data transfers employ Asynchronous 1394 block writes addressed to the Data_FIFO_Address. The actual assignments of memory addresses used to store these packets are implementation dependent.

The size of the 1394 block writes is limited to the Maximum Transmission Unit size of the 1394 speed between the Initiator and the Target.

Reliable data transmission depends on the fundamental 1394 guaranteed delivery mechanism of Asynchronous data transfer.

Data packets are required to be delivered in order.

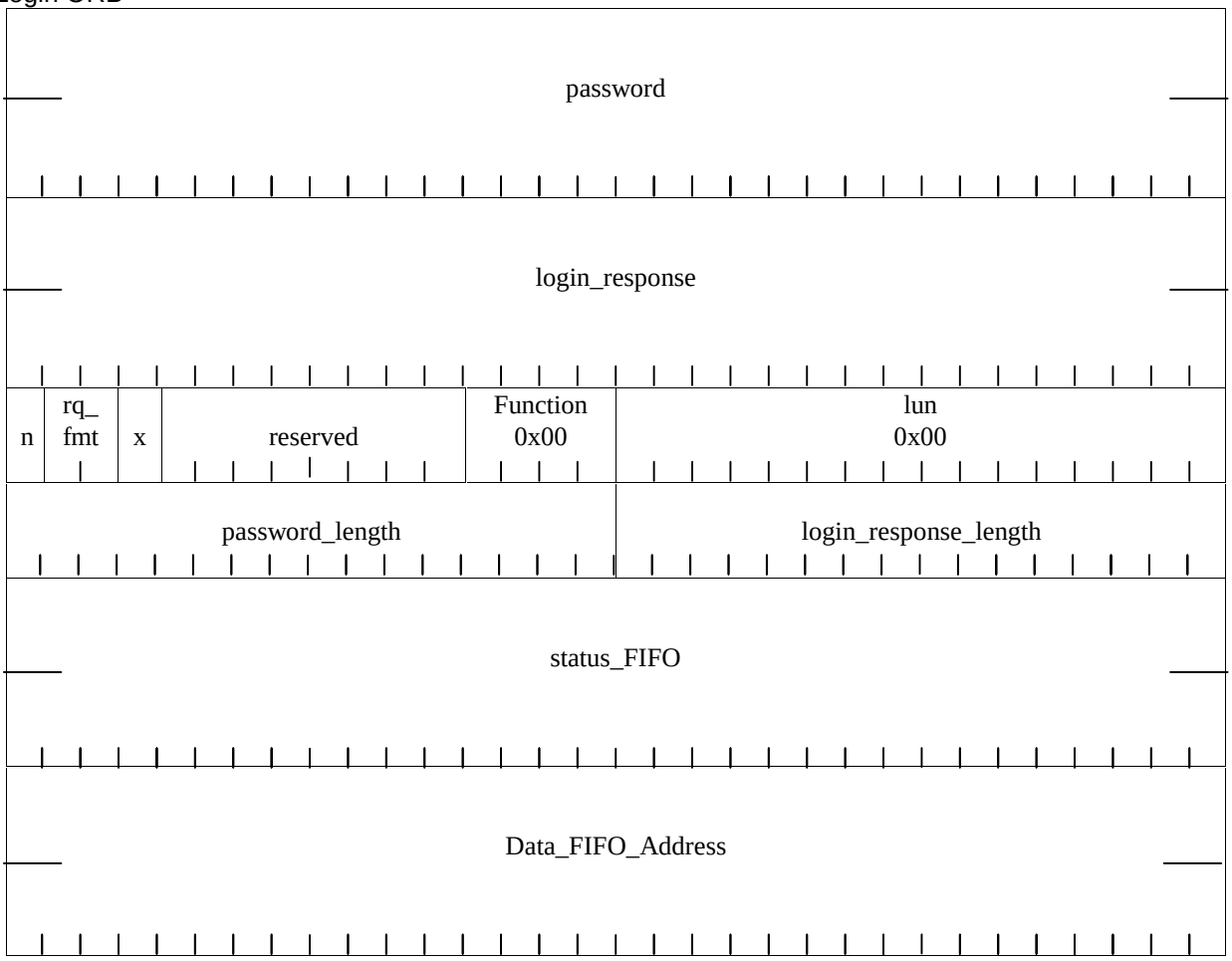
14. Login & Login Response

This section will specify the details of the Login Process.

The primary reasons for Login are access control, unsolicited status and the simple SBP-2 reconnect scheme.

1) The Initiator will discover the Target by reading the CSR & Configuration ROM space of devices on the 1394 bus. The Initiator can identify the Target through yet to be determined fields in registers or directories.
2) The Initiator will record the Target's GUID.
3) The Management_Agent register is also discovered at this time.

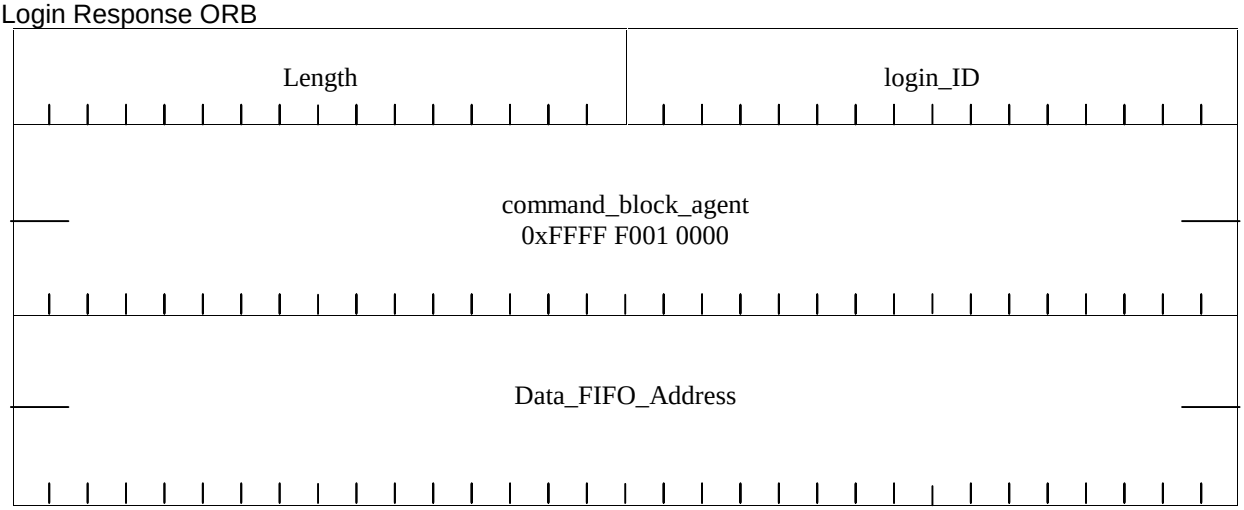
Login ORB



4) The Initiator will build the Login ORB with password.
5) The login_response address is the temporary memory address that the Target will use to send it's response to the Login.
6) The Status_Fifo address will remain static during the life of the Login.
7) The Data_FIFO_Address is used for Asynchronous Data Transfer.
8) The Initiator will write the address of the Login ORB to the Target's Management_Agent register. [Target shall monitor writes to this address or set up an Interrupt]

9) The 1394 speed (s100, s200, s400) of this write will determine the speed used for communication.

Login Response ORB



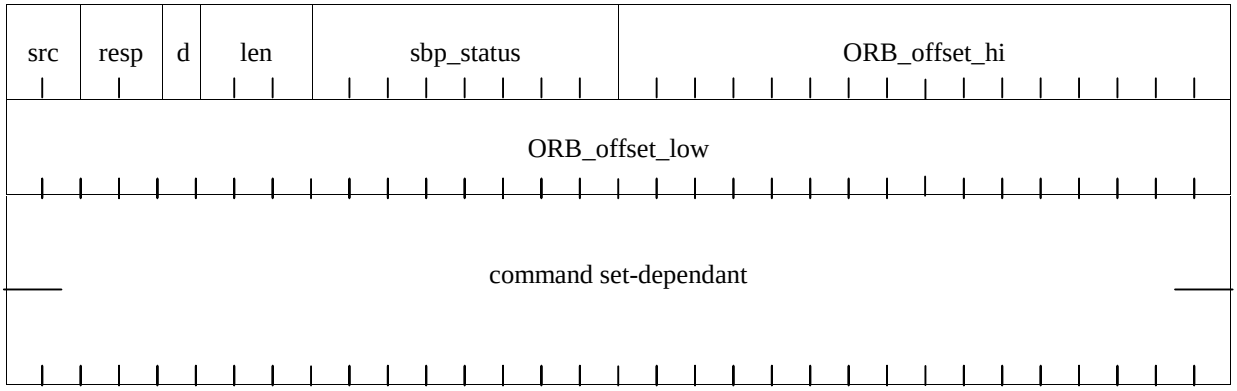
10) The Target will read the Login ORB.
11) The Target will read the Initiator's bus information block to discover the GUID.
12) The Target will validate this new Login by comparing the GUID against current login_descriptors. If this Initiator is already logged in the Login shall be rejected. If the Target only supports one Login and another device is logged in, the Login shall be rejected.
13) The Target will build a login_descriptor data structure that will be associated with this specific login.
14) The Target will store the Initiators GUID in the login_descriptor login_owner field.
15) The Target will build the Login Response ORB and fill in the login_ID. The login_ID is like a connection identifier that is unique across active Logins.
16) The Target will store the login_ID in the login_descriptor.
17) The command_block_agent address points to Configuration ROM.
18) The Data_FIFO_Address is used for Asynchronous Data Transfer.
19) Finally the Target writes the Login Response ORB to the login_response address.

[This profile requires that the Login ORB contain either the current or master password for the Login to be successful.] [Do we want passwords?]

15. Unsolicited Status

1. The Target may send Unsolicited status to the Initiator using the Status_Fifo address that the Target received during Login.
2. The Target shall use its Unsolicited_Status_Enable register to handshake this status block.
3. The Target can only store status when the Unsolicited_Status_Enable register is set to one.
4. After writing the status the Target will clear this register.
5. The Initiator may write a one to the Target's Unsolicited_Status_Enable to allow subsequent Unsolicited Status.
6. The Unsolicited_Status_Enable register is cleared at successful Login.

Status Block

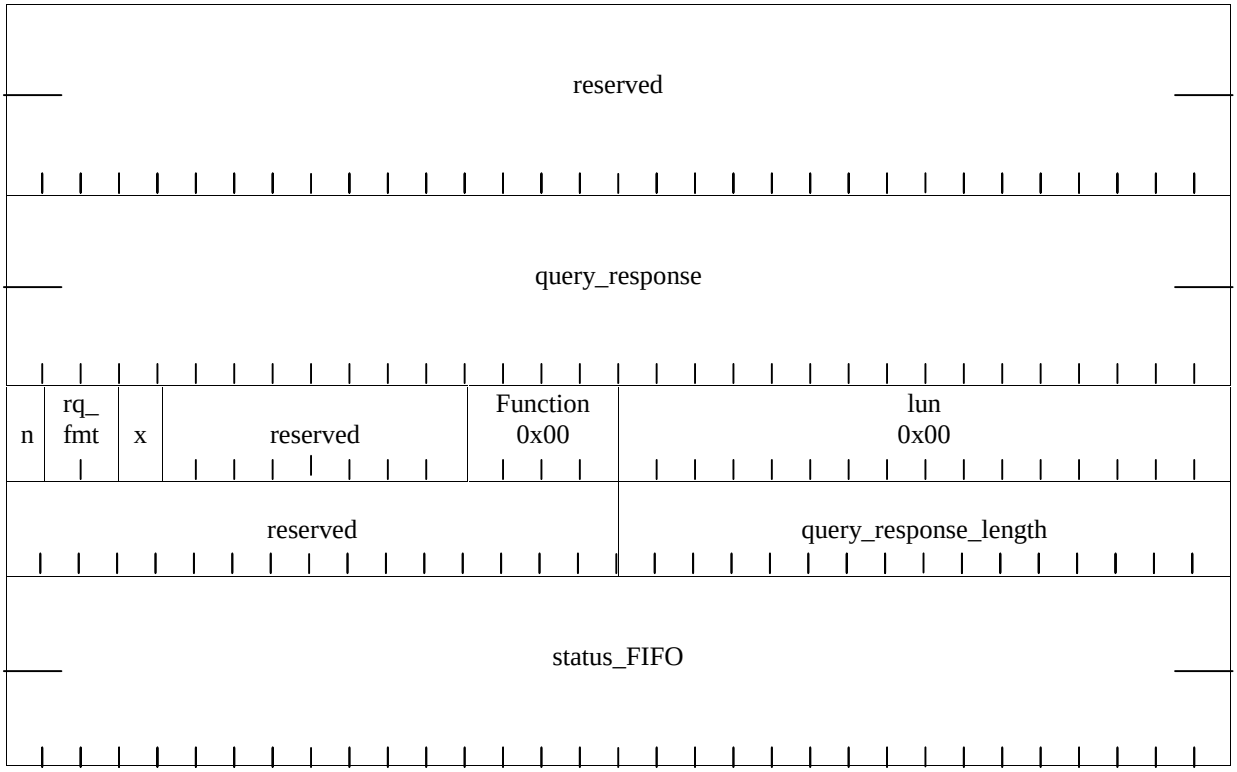


We may want to define status codes for the command set-dependant field. The type of status that could be reported using this mechanism is conditions such as paper out or paper jam.

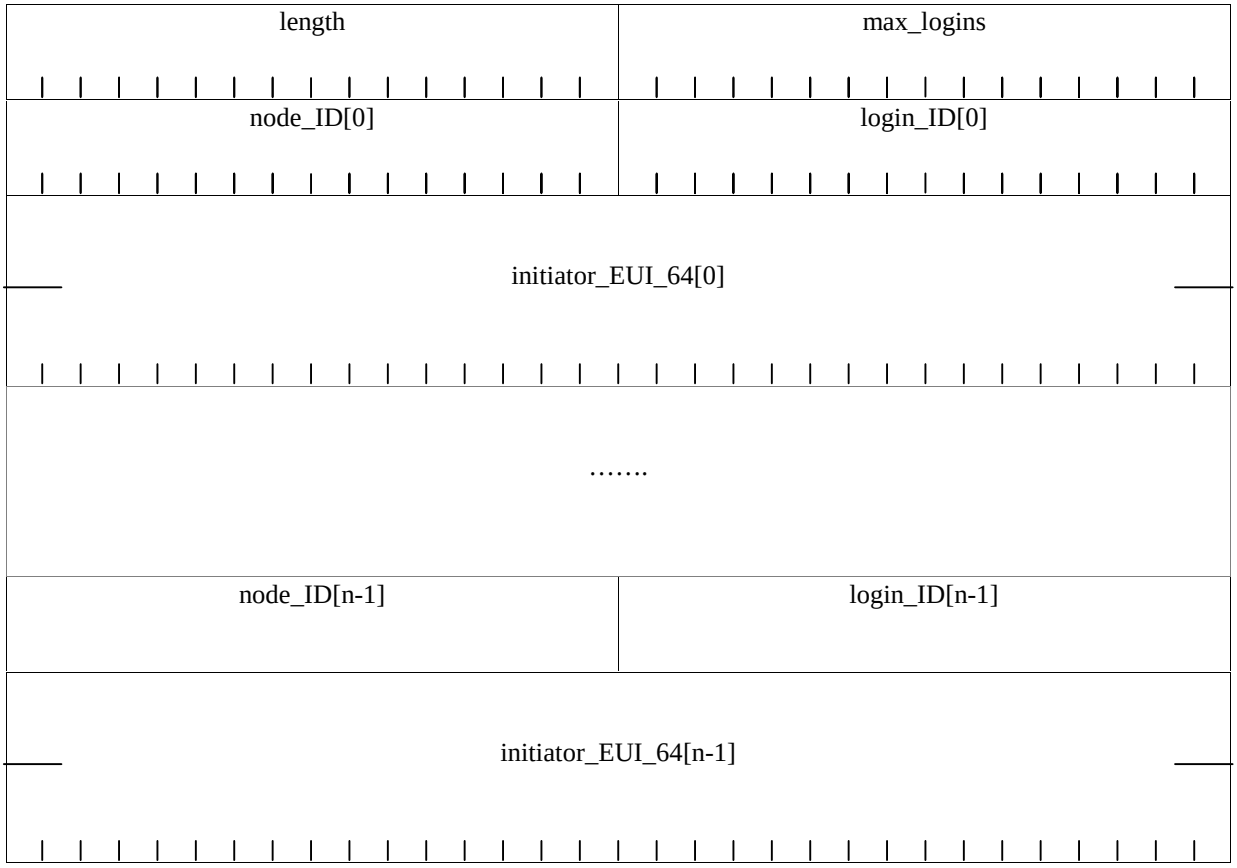
Need a table of applicable SBP-2 status codes here.

16. Query Logins

1. Inititiator may write the address of the Query Logins ORB to the Target Management_Agent register.
2. Target will fetch the ORB



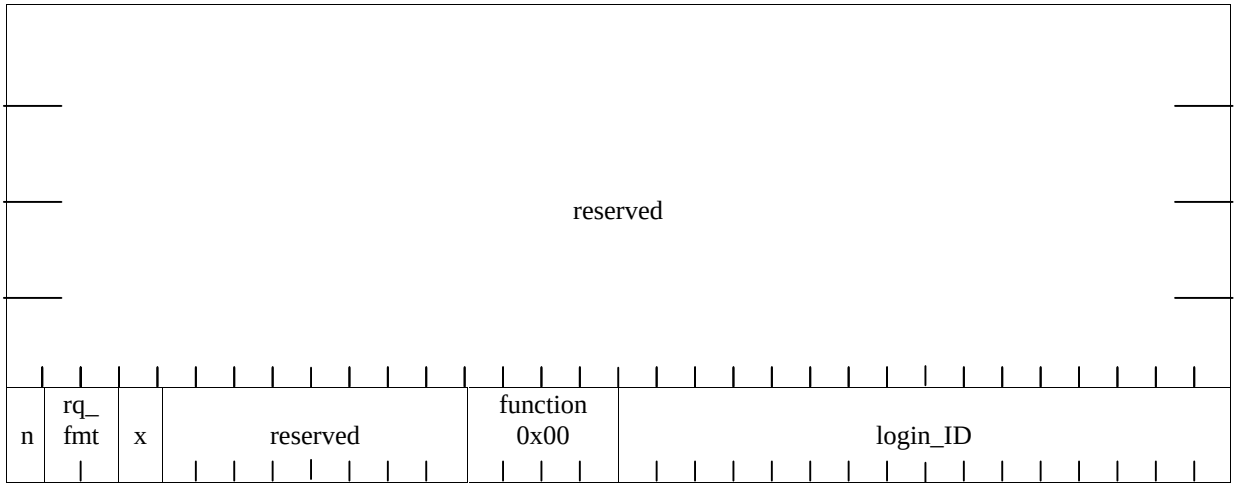
3. Target will write the Response to query_response address.

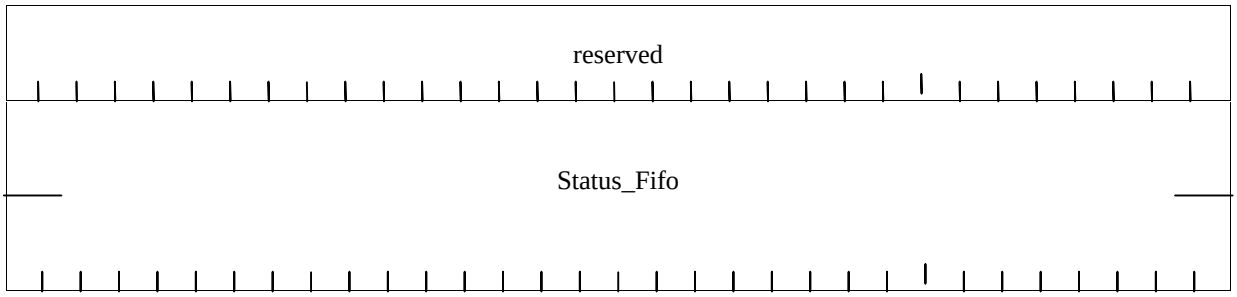


17. Reconnection

Reconnection after a Bus Reset will be accomplished using the Reconnect ORB.

Reconnect ORB





1. After a bus reset, the Initiator is required to re-discover the Target that it was Logged into by reading the Bus Information Blocks of nodes on the bus and searching for a matching GUID.
2. Once the Target is found the Initiator can write the address of the reconnect ORB to the Target's Management_Agent register.
3. The 1394 speed (s100, s200, s400) of this write will determine the speed used for communication.
4. The Initiator shall use the same login_ID that the Target provided at Login. The Target will fetch the reconnect ORB and read the Initiators Bus Information Block to verify that the Initiators GUID match the GUID established at Login.
5. The reconnect is completed when the Target writes status to the Status_Fifo. Note that this is a new separate reconnect Status_Fifo, which is not the same one established at Login.
6. After reconnect the original Status_Fifo is used for unsolicited status.
7. The original Login Status_Fifo address may have to be patched with the Initiators new node number and bus number.
8. The Data_FIFO_Addresses may also have to be patched with the new Node number and Bus number.

18. Logout

19. Higher Layer Protocols

Higher layer protocols could provide routing information, flow control and support for multiple logical channels.

Should the 1394 PWG specify a higher layer protocol?
Should the PWG define new packet headers?

Propose that we specify IEEE 1284.4 since it addresses all of the concerns above.

Issue

- If multiple protocols use the DFA method how can we tell if the data payload packet header is 1284.4 or if it is something else.

20. Error Recovery

Asynchronous Data Transmission Error Recovery

21. Fairness