ATLAPRec structure

#include <AppleTalk.h>

typedef struct **ATLAPRec** {

		Size	Offset	Description
ABCallType	abOpcode;	1	0	Type of call
short	abResult;	2	2	Result code
long	abUserReference;	4	4	For your use
LAPAdrBlock	lapAddress;	4	8	Destination or source node ID
short	lapReqCount;	2	12	Length of frame data or buffer size in bytes
short	lapActCount;	2	14	Number of frame data bytes actually received
Ptr	lapDataPtr;	4	16	Pointer to frame data or pointer to buffer
} **ATLAPRec**;		20		

typedef ATLAPRec ***ATLAPRecPtr**;
typedef ATLAPRec ****ATLAPRecHandle**;

Notes: When an ALAP frame is sent, the lapAddress field indicates the ID of the destination node. When an ALAP frame is received, lapAddress returns the ID of the source node. The lapAddress field also indicates the ALAP protocol type of the frame.

When an ALAP frame is sent, lapReqCount indicates the size of the frame data in bytes and lapDataPtr points to a buffer containing the frame data to be sent. When an ALAP frame is received, the lapDataPtr points to a buffer in which the incoming data can be stored and lapReqCount indicates the size of the buffer in bytes. The number of bytes actually sent or received is returned the the lapActCount field.

Each ALAP frame contains an 8-bit ALAP protocol type in the header. ALAP protocol types 128 through 255 are reserved for internal use by ALAP, hence the declaration:

typedef Byte ABByte; ALAP protocol type

Warning: Do not use ALAP protocol type values 1 and 2; they're reserved for use by DDP. Values 3 through 15 are reserved for internal use by Apple and also should not be used.

Using ALAP

Most programs will never need to call ALAP, because higher-level protocols will automatically call it as necessary. If you do want to send a frame directly via ALAP, cal the **LAPWrite** function. If you want to read ALAP frames, you have two choices:

- Call **LAPOpenProtocol** with NIL for protoPtr; this installs the default protocol handler provided by the **AppleTalk Manager**. Then call **LAPRead**to receive frames.
- Write your own protocol handler, and call **LAPOpenProtocol** to add it to the node's protocol handler table. The ALAP code will examine every incoming frame and send all those with the correct ALAP protocol type to

your protocol handler. When your program no longer wants to receive frames with a particular ALAP protocol type value, it can call **LAPCloseProtocol** to remove the corresponding protocol handler from the protocol handler table.