Inter-Office Memorandum

To	Alto Gateway Project	Date	July 31, 1978
From	David Boggs	Location	Palo Alto
Subject	How to install extra Ethernet Interfaces in an Alto	Organization	Parc

XEROX

Filed on: [Ivy]<Portola>ExtraEther.bravo

This memo describes how to install up to two extra Ethernet interfaces in an Alto. This can easily be done to any vintage Alto though these instructions are for an Alto II. The hardware configuration of your Alto determines which card slots and tasks you use, so this memo is not a complete recipe - you must understand what you are doing.

Mechanical considerations

An extra Ethernet interface may be installed in any spare processor slot, 15-20. There are no uncommitted connector mounting holes on the rear bulkhead, but the TRICON radial cable hole is the right size and will work for the first extra interface. Be sure to clearly label both ends of the cable.

Board modifications

An Ethernet board used in one of the extra positions needs several modifications. The ICmd & OCmd flip flop inputs must be disconnected from BUS[14-15]:

cut the trace at 35-2
cut the trace at 35-14,

and brought out to edge pins so that they can be set by jumpers:

add a wire from 35-2 to edge pin 98 (OCmd)
add a wire from 35-14 to edge pin 97 (ICmd).

The signal IBusy, which is brought out to an edge pin for debugging, collides with TaskA', so

cut the trace at edge pin 113.

To prevent the extra boards from responding to the emulator's ReadSerialNumber function, and to avoid driving the signal SIO from more than one place,

remove the 3205 at position 9.

If this is an Alto II, replace the following chips with schottky versions:

7402 at position 40
7404 at position 55
74157 at position 48
7438s at positions 14, 15, 24, 25, 26.

A modified board will work in a normal Ethernet slot if you replace the 3205 at position 9 and jumper pin 14-98 to 14-95 and pin 14-97 to 14-94 on the backplane.

Backplane modifications

An Ethernet board needs some signals which are not present on the standard processor bus slots. These are available on the corresponding pins of slot 14, the standard Ethernet.

SysClk'	12
AuSysClk	72
_KData'	111
EmAct'	99
SWakMRT'	68

In addition, two BUS bits must be connected to the Cmd flip flops, and a task must be assigned by connecting the board's Active and Wakeup signals. These are discussed below.

Note that SReset' and EStop are not wired on extra interfaces. SReset is the signal which boots the machine, and it is sufficient for the standard Ethernet to yank on it; besides, SIO decoding on the extra boards is disabled. EStop is the signal which stops the clocks for one cycle to fix a long path in the interface. Installing Schottky chips in the path makes it unnecessary to do this.

Host addresses

The host address logic in an extra Ethernet interface is disabled by removing the 3205, so the SIO instruction returns the address set by the jumpers on the standard Ethernet interface in slot 14. Host jumpers on the extra slots are not required.

Tasks, SIO bits, and Page 1 locations

The choice of task for an extra Ethernet interface is invisible to the emulator level program. An active interface consumes about 15% of an Alto, which is low enough that any of the four uncommitted tasks available on the backplane will work. Pick one of them and wire its wakeup and active pins on the control board (slot 11) to EtherWakeup' (pin 103) and EtherActive' (pin 100) on the extra Ethernet board. The table below gives the pin numbers on the control board for the uncommitted tasks.

Task	Wakeup'	Active'
1	113	119
2	58	52
5	60	102
6	104	101

The microcode for the extra interfaces use page 1 locations 630-640B and 642-652B in the same way that the standard Ethernet uses 600-610B. The extra interfaces may be assigned different host addresses than the standard one by putting different numbers in 640B and 652B, but as mentioned above, SIO returns the address set on the backplane of the standard interface so you must invent a new way to get the additional addresses. Unless there is a compelling reason, I recommend that additional interfaces use the same host address as the standard one.

The emulator task signals an Ether task by placing a value on BUS and executing the SIO emulator function. Each Ethernet interface checks two BUS bits during an SIO and wakes up its task if either bit is one. The task then performs some action which ends up modifying its page 1 locations. Thus the software must know the correspondence between SIO bits and page 1 locations. I recommend the following correspondence:

SIO bits	page 1	
14 & 15	600-610B	(standard Ethernet interface)
12 & 13	630-640B	
10 & 11	642-651B	

where the MSB of the pair sets the ICmd FF by being wired to pin 97, and the LSB sets the OCmd FF by being wired to pin 98. The MESA and BCPL PUP packages assume this; if you do it differently you forfeit compatibility. BUS[0-15] are on pins 80-95.

Microcode

Files ExtraEther1.mu and ExtraEther2.mu contain copies of the Ether microcode which use page 1 locations 630-640B and 642-652B respectively. These files do not define task numbers or R-registers to be used. File ExtraEther.mu is an example of how to do this, assigning tasks 2 and 3, and registers 14-17B, and adding enough other definitions to make a stand-alone ram image for two extra Ethernets. These files are stored in [Ivy]<Portola>GatewayMc.dm

Note that the registers must be in the first group of 32 since the Ether hardware can't be ram-related (function and bus sources collide with the ram). This is a problem for MESA, since only one R-register is available. Another one can be freed by rewriting the memory refresh task to eliminate its use of ClockTemp. To get the next two, the cursor must be sacrificed. This involves deleting all references to CurX and CurData from the MRT, Cursor, and DVT tasks.

Revision History

July 20, 1978

First release.