# IBM PCI Pit/Pit-Phy/Olympic CHIPSET BASED TOKEN RING CARDS README

Release 0.2.0 - Release
June 8th 1999 Peter De Schrijver & Mike Phillips
Release 0.9.C - Release
April 18th 2001 Mike Phillips

Thanks:
Erik De Cock, Adrian Bridgett and Frank Fiene for their
patience and testing.
Donald Champion for the cardbus support
Kyle Lucke for the dma api changes.
Jonathon Bitner for hardware support.
Everybody on linux-tr for their continued support.

## Options:

The driver accepts four options: ringspeed, pkt_buf_sz,
message_level and network_monitor.

These options can be specified differently for each card found.

ringspeed: Has one of three settings 0 (default), 4 or 16. 0 will
make the card autosense the ringspeed and join at the appropriate speed,
this will be the default option for most people. 4 or 16 allow you to
explicitly force the card to operate at a certain speed. The card will fail
if you try to insert it at the wrong speed. (Although some hubs will allow
this so be *very* careful). The main purpose for explicitly setting the ring
speed is for when the card is first on the ring. In autosense mode, if the card
cannot detect any active monitors on the ring it will not open, so you must
re-init the card at the appropriate speed. Unfortunately at present the only
way of doing this is rmmod and insmod which is a bit tough if it is compiled
in the kernel.

pkt_buf_sz: This is this initial receive buffer allocation size. This will
default to 4096 if no value is entered. You may increase performance of the
driver by setting this to a value larger than the network packet size, although
the driver now re-sizes buffers based on MTU settings as well.

message_level: Controls level of messages created by the driver. Defaults to 0:
which only displays start-up and critical messages. Presently any non-zero
value will display all soft messages as well. NB This does not turn
debugging messages on, that must be done by modified the source code.

network_monitor: Any non-zero value will provide a quasi network monitoring
mode. All unexpected MAC frames (beaconing etc.) will be received
by the driver and the source and destination addresses printed.
Also an entry will be added in /proc/net called olympic_tr%d, where tr%d
is the registered device name, i.e tr0, tr1, etc. This displays low
level information about the configuration of the ring and the adapter.
This feature has been designed for network administrators to assist in
the diagnosis of network / ring problems. (This used to OLYMPIC_NETWORK_MONITOR,
but has now changed to allow each adapter to be configured differently and
to alleviate the necessity to re-compile olympic to turn the option on).

Multi-card:

The driver will detect multiple cards and will work with shared interrupts, each card is assigned the next token ring device, i.e. tr0 , tr1, tr2. The driver should also happily reside in the system with other drivers. It has been tested with ibmtr.c running, and I personally have had one Olicom PCI card and two IBM olympic cards (all on the same interrupt), all running together.

Variable MTU size:

The driver can handle a MTU size upto either 4500 or 18000 depending upon ring speed. The driver also changes the size of the receive buffers as part of the mtu re-sizing, so if you set mtu = 18000, you will need to be able to allocate 16 * (sk_buff with 18000 buffer size) call it 18500 bytes per ring position = 296,000 bytes of memory space, plus of course anything necessary for the tx sk_buff's. Remember this is per card, so if you are building routers, gateway's etc, you could start to use a lot of memory real fast.

6/8/99 Peter De Schrijver and Mike Phillips