Linux と NIC(RTL8168d/8111d) におけるパケット送信処理の解析資料

2013/11/6

増田陽介

1 はじめに

現在増田の研究テーマである，Mint における割り込みルーティングの変更による NIC の移譲を実現するために，改修が必要な箇所について調査している．資料＜ No.237-03 ＞で，パケット送信時の NIC と NIC ドライバの動作について述べた．この際，処理の全体像が十分に説明できていなかった．本資料では，まずパケット送信処理の概要を示す．その後，パケット送信処理に使用するデータ構造やパケット送信処理について詳細に説明する．

本資料の構成を以下に示す．

(1) はじめに
(2) 調査環境
(3) 送信処理の概要
(4) パケット送受信時に使用されるデータ構造
(5) NIC ドライバが使用する I/O 命令
(6) 送信処理の詳細
(7) おわりに

2 調査環境

調査環境を表 1 に示す．

表 1　調査環境

項目名	環境
OS	Fedora 14
カーネル	Linux カーネル 3.0.8 64bit
CPU	Intel(R) Core(TM) Core i7-870 @ 2.93GHz
メモリ	2GB
NIC	RTL8168d/8111d
NIC デバイスドライバ	RTL8169

3 送信処理の概要

3.1 送信処理における NIC ドライバの役割

送信処理における NIC ドライバの役割について説明する．NIC ドライバの役割は，ソケットバッファに格納されているパケットをカプセル化し，NIC にパケットの送信を依頼することである．

送信処理において，NIC ドライバが主に使用する変数を以下に示す．ソケットバッファ，送信ディスクリプタ，および送信バッファに関しては 4 章で説明している．また，mmio_addr を用いた I/O 命令に関しては 5 章で説明している．

(1) ソケットバッファ
送信パケットが格納されている構造体である．NIC ドライバのパケット送信処理関数が上位層に呼び出される際に，引数として上位層から渡される．

(2) TXDescArray
送信ディスクリプタの先頭アドレス．送信ディスクリプタは NIC ドライバと NIC の間で情報をやり取りする際に使用する構造体である．

(3) tx_skb
送信バッファの先頭アドレス．送信バッファは．パケットのデータ長とソケットバッファを持つ構造体である．NIC ドライバが，パケットが複写されている DMA 領域と，ソケットバッファを解放する際に使用する．

(4) mmio_addr
NIC のレジスタがマッピングされている仮想アドレス空間の先頭アドレス．NIC ドライバが NIC へ I/O 命令を通知する際に使用する．

3.2 送信処理における NIC の役割

送信処理における NIC の役割について説明する．NIC の役割は，NIC ドライバによって処理されたパケットを，DMA を用いて取得し送信することである．その際に，送信ディスクリプタを更新し，OS に送信完了割り込みを通知する．送信処理において，NIC が主に使用するレジスタを以下に示す．

(1) TPPoll(Transmit Priority Polling)
送信処理を制御する NIC のレジスタ．NIC ドライバがこのレジスタに 1 を書き込むことで，NIC は送信処理を開始する．送信できるパケットを全て送信し終えると，NIC はこのレジスタをクリアし，送信処理を終了する．

(2) TNPDS(Transmit Normal Priority Descriptors Start address)
実メモリ空間に配置されている送信ディスクリプタの先頭アドレスを持つレジスタ．

3.3 送信処理の流れ

パケット送信処理おける，NIC と NIC ドライバが行う処理の概要を図 1 に示し，以下で説明する．送信処理の詳細は，6 章で説明している．

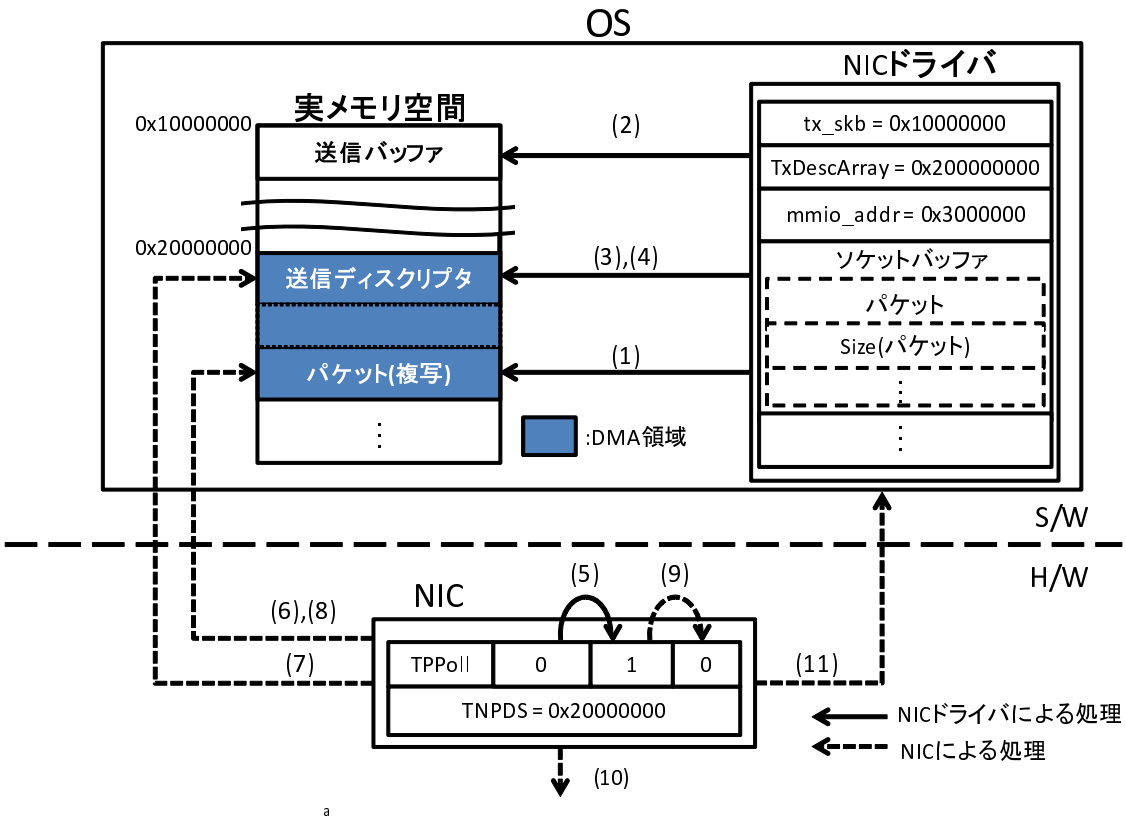

図 1　送信処理の概要

(1) パケットを DMA 領域に複写
NIC ドライバは，ソケットバッファに格納されているパケットを，DMA 領域に複写する．

(2)　送信ディスクリプタにパケットのアドレスを書き込み
NIC ドライバは，(1) で複写したパケットの先頭アドレスを送信ディスクリプタに書き込む．

(3) 送信ディスクリプタを更新
NIC ドライバは，NIC が送信処理に使用する情報を，送信ディスクリプタに書き込む．

(4) 送信バッファにソケットバッファの情報を書き込み
NIC ドライバは，送信バッファに，パケットのデータ長とソケットバッファの先頭アドレスを書き込む．送信バッファは，送信処理には使用されない．NIC ドライバが，パケットを複写した DMA 領域とソケットバッファを解放する際に使用する．

(5) NIC に送信処理を依頼

NIC ドライバは，MMIO を用いて NIC の TPPoll レジスタを 1 にすることで，NIC にパケット送信処理の実行を依頼する．

(6) 送信ディスクリプタを参照
NIC は TNPDS が持つアドレスを元に，送信ディスクリプタを参照する．TNPDS は，送信ディスクリプタの先頭アドレスを持つ NIC のレジスタである．

(7) パケットを取得
NIC は，送信ディスクリプタに記載されているアドレスにアクセスし，パケットを取得する．

(8) パケットを送信
NIC は，取得したパケットを送信する．

(9) 割り込みを通知
NIC は，パケットの送信を完了したことを通知するために，OS に割り込みを通知する．

(10) 送信ディスクリプタを更新
NIC は，送信ディスクリプタを送信済みのものに更新する．

(11) 送信終了処理
NIC は，TPPoll を 0 にし，送信処理を終了する．

4 パケット送信時に使用されるデータ構造

パケット送信時に，NIC および NIC ドライバが使用するデータ構造について説明する．データ構造と実メモリ空間との関係図を図 2 に示す．以降で，それぞれのデータ構造について説明する，

(1) ソケットバッファ
ソケットバッファは上位層と NIC ドライバでやり取りされるデータであり，NIC ドライバの送信処理関数に引数として渡される．NIC ドライバは，ソケットバッファに格納されているパケットを送信するよう NIC に依頼する．ソケットバッファの概要を図 3 に示し，以下で説明する．
head はソケットバッファの先頭を示し．tail はデータの終端を示す．data は，バッファに格納されているデータの先頭を示す．送信時に，NIC ドライバにパケットとして DMA 領域に複写されるのはこの data 部である．end はソケットバッファの終端を示す．

(2) 送信バッファ
送信処理に使用したソケットバッファ本体のアドレスとソケットバッファが持つパケットのデータ長を持つ構造体．以下のように宣言されている．

```
root/drivers/net/r8169.c
638 struct ring_info tx_skb[NUM_TX_DESC];   /* Tx data buffers */
```

構造を以下に示す．

```
struct ring_info {
```

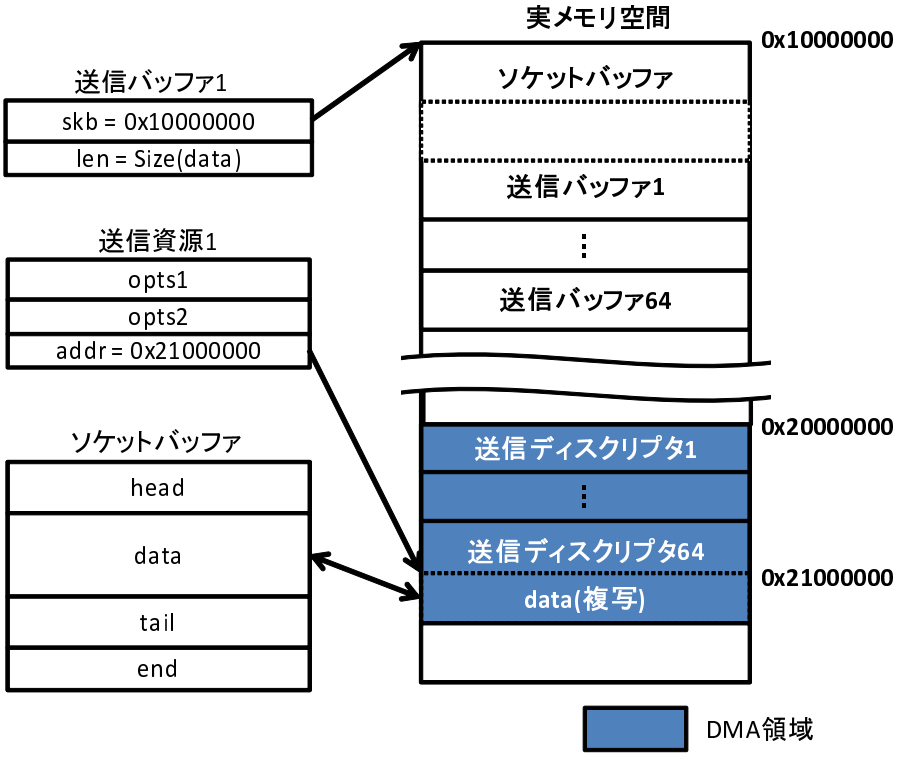

図 2　パケット送信時に使用されるデータ構造

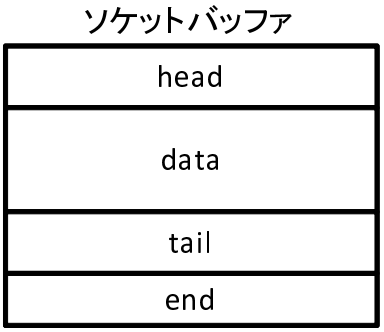

図 3　ソケットバッファの構造

```
        struct sk_buff  *skb;
        u32             len;
        u8              __pad[sizeof(void *) - sizeof(u32)];
};
```

送信バッファは，送信処理時には使用されず，NIC ドライバの割り込み処理時に使用される．skb は，NIC ドライバに引数として渡されたソケットバッファ本体の先頭アドレスを持つ．ソ

ケットバッファの先頭アドレスはmソケットバッファを解放する関数である dev_free_skb() の引数として使用される．len は，DMA 領域に複写されたパケットのデータ長を持つ．パケットのデータ長は，パケットを解放する関数である rtl8169_unmap_tx_skb() の引数として使用される．

(3) 送信ディスクリプタ

送信ディスクリプタは．NIC と NIC ドライバがパケット送信時に使用するデータ構造であり，配列として DMA 領域に確保されている．送信ディスクリプタの配列はリングバッファであり，エントリ数は 64 個である．

送信ディスクリプタの構成を以下に示し，説明する．

```
struct TxDesc{
    __le32 opts1;
    __le32 opts2;
    __le64 addr;
}
```

opts1 は，送信ディスクリプタに関する情報と，パケットに関する情報を持つ．opts2 は，VLAN(Virtial LAN) の設定情報を持つ．addr は，DMA 領域上に複写されたパケットの先頭アドレスを持つ．

変数 opts1 には 2 つの状態がある．未送信パケットを持つ状態と，送信済みパケットを持つ状態である．各変数の構造を以下に示す．

（A）未送信パケットを持つ状態

未送信パケットを持つ TxDesc の構造を図 4 に示し，説明する．

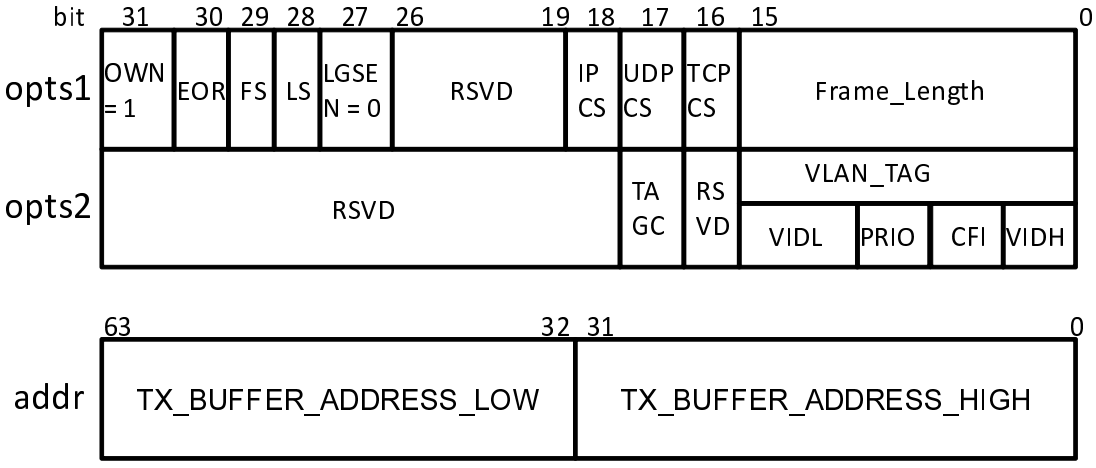

図 4　未送信パケットを持つ送信ディスクリプタの構造

(a) opts1

(i) OWN(bit31)
送信ディスクリプタの状態を表すビット．OWN ビットが 0 の場合，送信済みパケットを持つ送信ディスクリプタであることを示す．OWN ビットが 1 の場合，未送信パケットを持つ送信ディスクリプタであることを示す.

(ii) EOR(bit30)
送信ディスクリプタがリングバッファの終端 (End Of Descriptor Ring) にあるものかを判別するビット．EOR が 1 の場合，送信ディスクリプタはリングバッファの終端にある.

(iii) FS(bit29)
このビットが 1 の場合，送信ディスクリプタに割り当てられているパケットが分割されたパケットの先頭 (First Segment) であることを示す.

(iv) LS(bit28)
このビットが 1 の場合，送信ディスクリプタに割り当てられているパケットが分割されたパケットの終端 (Last Segment) であることを示す.

(v) RSVD(bit26〜19)
使用されていない領域.

(vi) IPCS(bit18)
IP checksum offload を実行するか否かを示すビット．このビットが 1 の場合, NIC は NIC ドライバの代わりに IP チェックサムの算出を行う.

(vii) UDPCS(bit17)
UDP checksum offload を実行するか否かを示すビット．このビットが 1 の場合，NIC は NIC ドライバの代わりに UDP チェックサムの算出を行う.

(viii) TCPCS(bit16)
TCP checksum offload を実行するか否かを示すビット．このビットが 1 の場合，NIC は NIC ドライバの代わりに TCP チェックサムの算出を行う.

(ix) Frame_Lnegth(bit15〜0)
パケットのデータ長を持つ領域.

(b) opts2

(i) RSVD(bit31〜18)
使用されていない領域.

(ii) TAGC(bit17)
パケットがタグ VLAN に対応しているか否かを示すビット．このビットが 1 の場合，NIC はパケットに VLAN_TAG の情報を付与する.

(iii) RSVD(bit16)
使用されていない領域.

(iv) VLAN_TAG(bit15〜0)
VLAN の識別に使用するタグ情報を持つ領域.

（B）送信済みパケットを持つ状態

送信済みパケットを持つ TxDesc の構造を図 5 に示し，説明する．なお，opts2 と addr に関しては未送信パケットを持つ状態と同様であるため省略する．未送信パケットを持つ送信

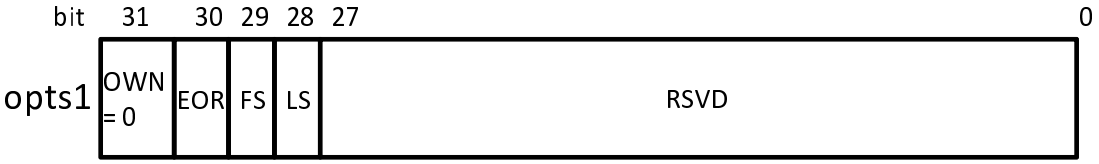

図 5　送信済みパケットを持つ送信ディスクリプタの構造

ディスクリプタとの相違点として，OWN ビットが 0 に書き換えられている．これは，送信済みパケットを持つ送信ディスクリプタであることを示す．OWN ビットが 0 である送信ディスクリプタは，NIC ドライバにより初期化される．EOR,FS,LS は，未送信パケットを持つ状態で説明したものと同様である．27 ビットから 0 ビットは，RSVD 領域に書き換えられている．

5　NIC ドライバが使用する I/O 命令

NIC ドライバが用いる I/O 命令について説明する．NIC ドライバは，NIC への I/O 命令にメモリマップド I/O を用いる．メモリマップド I/O は仮想メモリ上のマッピングを用いて NIC のレジスタへアクセス手法である．

具体的には以下のマクロを用いて，NIC にアクセスする．

```
root/drivers/net/r8169.c
93 /* write/read MMIO register */
94 #define RTL_W8(reg, val8)       writeb ((val8), ioaddr + (reg))
95 #define RTL_W16(reg, val16)     writew ((val16), ioaddr + (reg))
96 #define RTL_W32(reg, val32)     writel ((val32), ioaddr + (reg))
97 #define RTL_R8(reg)             readb (ioaddr + (reg))
98 #define RTL_R16(reg)            readw (ioaddr + (reg))
99 #define RTL_R32(reg)            readl (ioaddr + (reg))
```

6　送信処理の詳細

6.1　NIC が持つ情報

NIC が持つ情報のうち，送信処理時に使用されるものを以下に示す．図 6 に示し，以下で説明する．

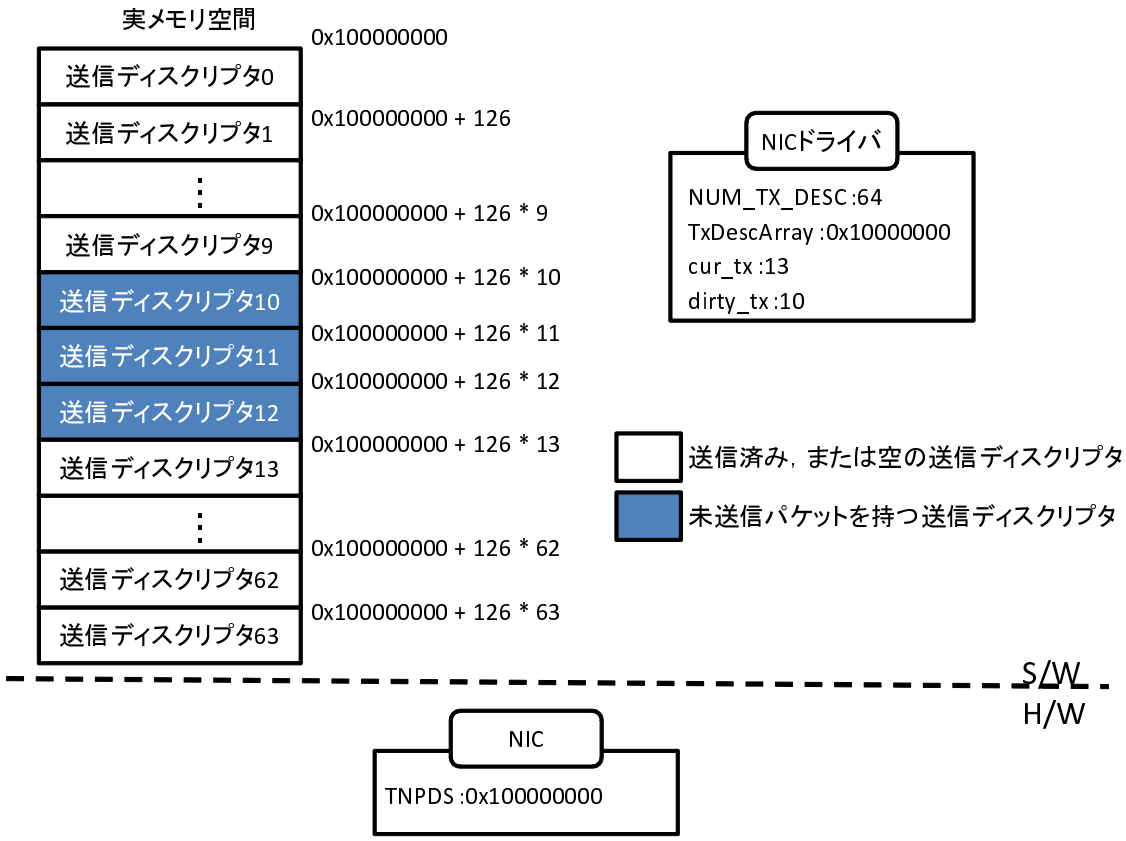

図 6　送信に使用される情報

(1) TPPoll(Transmit Priority Polling)
NIC の送信処理を制御するレジスタ．NIC ドライバがこのレジスタに 1 を書き込むことで，NIC は送信処理を開始する．送信できるパケットを全て送信し終えると，NIC はこのレジスタをクリアし，送信処理を終了する．

(2) TNPDS(Transmit Normal Priority Descriptors Start address)
実メモリ空間に配置されている送信ディスクリプタ配列の先頭アドレスを持つレジスタ．

6.2 NIC ドライバが持つ情報

NIC ドライバが持つ情報のうち，送信処理時に使用されるものを以下に示す．

(1) u32 cur_tx
NIC ドライバが起動してから使用した送信ディスクリプタの総数を持つ．

(2) u32 dirty_tx
NIC ドライバが送信処理に使用した後，NIC によって処理された送信ディスクリプタの総数を持つ．

(3) NUM_TX_DESC
送信ディスクリプタの総数である 64 を持つ．

(4) TxDesc TxDescArray
実メモリ空間に配置されている送信ディスクリプタ配列の先頭アドレスを持つ．

6.3 送信処理の詳細

(1) 送信可能か確認
NIC ドライバは，上位層からソケットバッファを受けとると送信ディスクリプタの数が足りているか否かを確認する．
送信ディスクリプタの数が足りている場合を図 7 に示し，送信ディスクリプタの数が足りていない場合を図 8 に示す．それぞれの場合について以下で説明する．

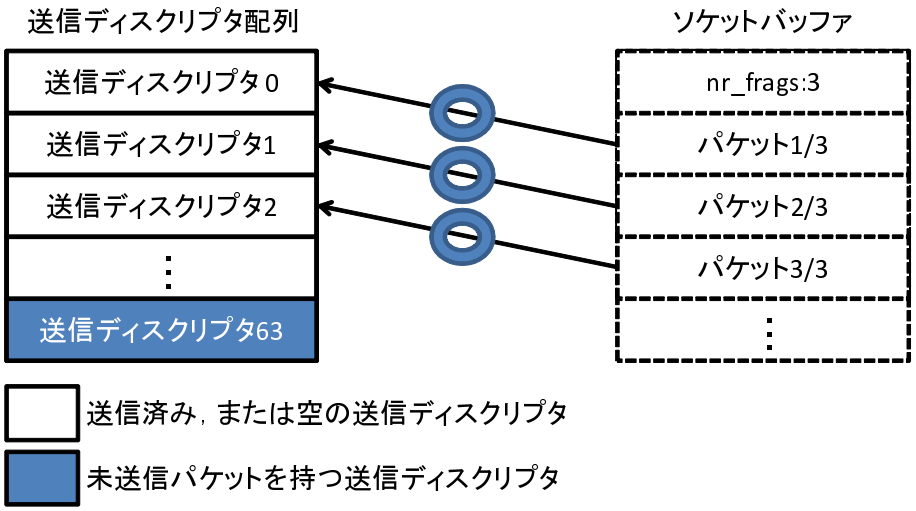

図 7　空き送信ディスクリプタが足りている場合

ソケットバッファには，パケットが分割されて格納されている場合がある．この際，分割されているパケットの数だけ送信ディスクリプタが必要になる．送信ディスクリプタの数が足りていな

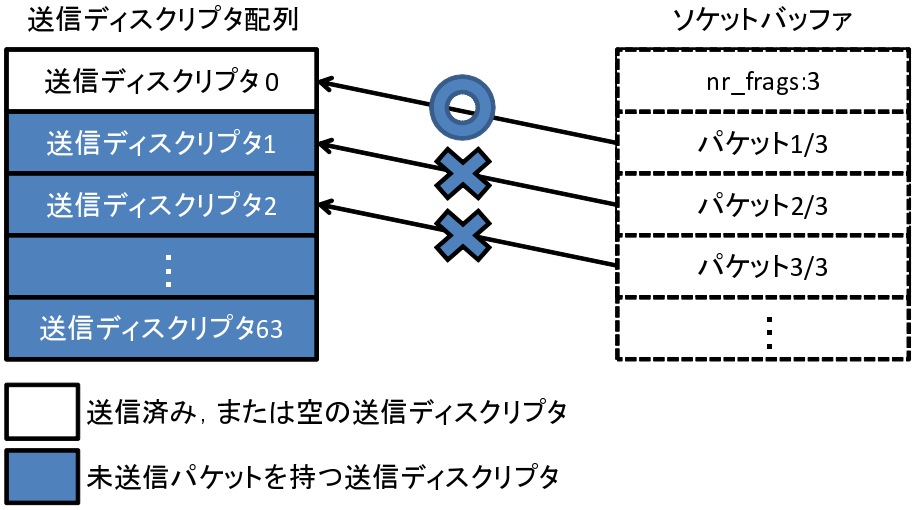

図 8　空き送信ディスクリプタが足りていない場合

い場合，NIC は送信処理を中止する．
以下に送信ディスクリプタの数が足りているか確認するコードを示し，説明する．

```
root/drivers/net/r8169.c
61 #define TX_BUFFS_AVAIL(tp) \
62 (tp->dirty_tx + NUM_TX_DESC - tp->cur_tx - 1)

4701  if (unlikely(TX_BUFFS_AVAIL(tp) < skb_shinfo(skb)->nr_frags)) {
4702          netif_err(tp, drv, dev, "BUG! Tx Ring full \
when queue awake!\n");
4703          goto err_stop_0;
4704  }
```

TX_BUFFS_AVAIL は，空き送信ディスクリプタの数を算出するマクロである．nr_frags は，ソケットバッファが持つ分割パケットの数が格納されている変数である．TX_BUFFS_AVAIL で算出した空き送信ディスクリプタの数が，nr_frags よりも下回っていた場合，NIC は送信処理を終了する．

(2) 送信ディスクリプタのエントリを算出
NIC ドライバが送信ディスクリプタのエントリを算出するコードを以下に示し，説明する．

```
root/drivers/net/r8169.c
4692  unsigned int entry = tp->cur_tx % NUM_TX_DESC;
4693  struct TxDesc *txd = tp->TxDescArray + entry;
```

cur_tx は，NIC ドライバが使用した送信ディスクリプタの総数である．NUM_TX_DESC はマクロであり，送信ディスクリプタのエントリの総数である 64 である．送信ディスクリプタはリングバッファとなっているため，cur_tx と NUM_TX_DESC との剰余は，最後に NIC ドライバが使用した送信ディスクリプタの次のエントリを指す．entry と送信ディスクリプタ配列の先頭アドレス TxDescArray を加算することで，次に使用する送信ディスクリプタの先頭アドレスを求める．

(3) パケットを DMA 領域に複写
ソケットバッファにはパケットが格納されている．NIC ドライバはこのパケットを DMA 領域に複写する．これは，NIC が DMA を用いてパケットを取得するためである．

(4) 送信バッファに書き込み
送信バッファに，複写したパケットの大きさとソケットバッファの先頭アドレスを格納する．送信バッファは送信処理には使用されず，送信完了後の割り込み処理時に，複写したパケットとソケットバッファを開放するために使用される．

(5) 送信ディスクリプタを更新
NIC ドライバは，空の送信ディスクリプタにパケットの情報を書き込む．これにより，送信ディスクリプタは更新され，未送信パケットを持つ状態になる．

(6) NIC に送信要求
5 章で示した I/O 命令を用いて NIC に送信要求を通知する．具体的には以下のコードにより通知する．

```
root/drivers/net/r8169.c
270  TxPoll   = 0x38,

405  NPQ      = 0x40,  /* Poll cmd on the low prio queue */

4747 RTL_W8(TxPoll, NPQ);
```

TxPoll は NIC の送信処理を制御する TPPoll レジスタのオフセットである．TPPoll に，NPQ の値が書き込まれることで NIC は送信処理を開始する．

(7) 参照する送信ディスクリプタのエントリを決定
NIC と NIC ドライバは，使用する送信ディスクリプタのエントリを個別に管理している．NIC が使用する送信ディスクリプタのエントリは，カウンタによって管理されており，これを送信ディスクリプタカウンタと呼ぶ．送信ディスクリプタカウンタについて図 9 に示し，以下で説明する．
NIC は送信時に，送信ディスクリプタを参照しパケットが複写されている送信ディスクリプタの

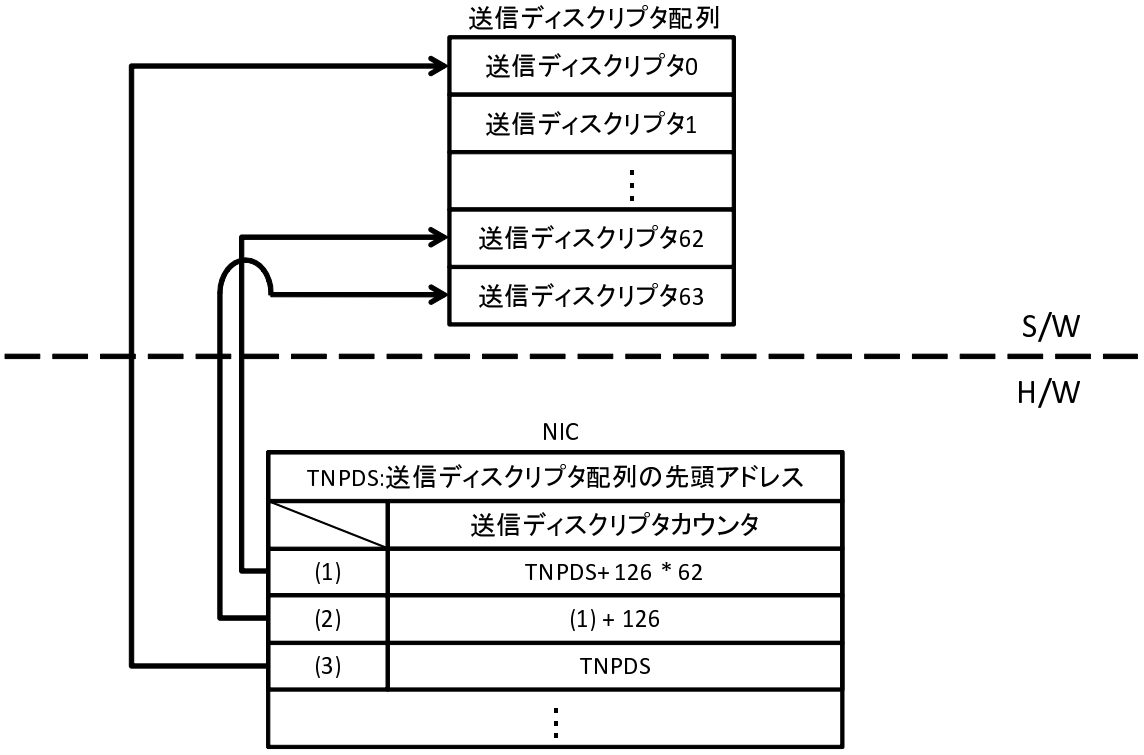

図 9　送信ディスクリプタカウンタを用いた送信ディスクリプタのエントリの決定

アドレスを得る．この際，参照する送信ディスクリプタのエントリは送信ディスクリプタカウンタによって管理されている．送信ディスクリプタカウンタは，NIC が送信ディスクリプタを参照するたびに 126 バイト加算され，次の送信ディスクリプタを指す．送信ディスクリプタ配列はリングバッファになっている．このため，送信ディスクリプタ配列の末尾のエントリをさした後，送信ディスクリプタカウンタは，送信ディスクリプタ配列の先頭アドレスで初期化される．

(8) 送信ディスクリプタを参照

(9) ソケットバッファを取得

NIC が送信ディスクリプタを参照し，パケットを取得する流れを図 10 に示し，以下で説明する．

(1) で，NIC は送信ディスクリプタカウンタのアドレスにある送信ディスクリプタ 0 を参照する．送信ディスクリプタ 0 は DMA 領域に複写されているパケットの先頭アドレスを持つ．NIC は送信ディスクリプタ 0 を参照後，DMA を用いてパケット (複写) を取得する．

(10) 送信

NIC は取得したパケットを送信する．

(11) 送信ディスクリプタを更新

送信ディスクリプタに格納されているパケットを送信した後，NIC は送信ディスクリプタを送信済みに更新する．送信済みの送信ディスクリプタは，NIC ドライバの割り込み処理中で初期化さ

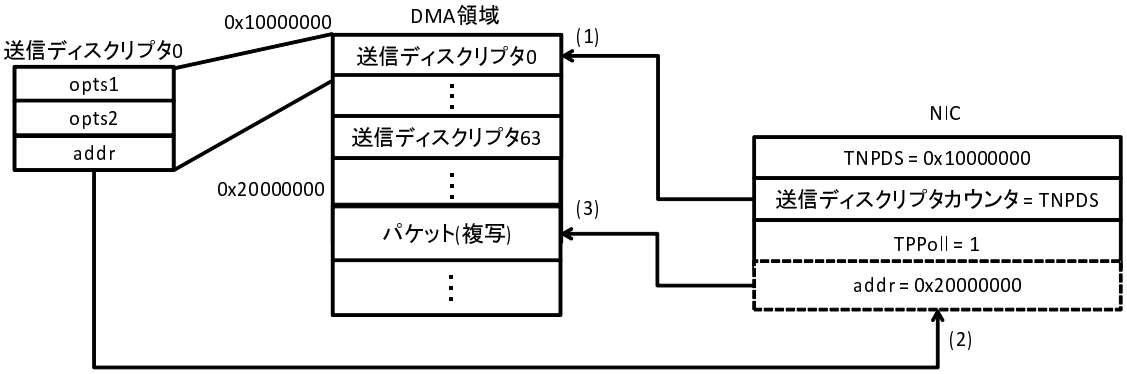

図 10　送信ディスクリプタを参照し，パケットを取得する流れ

れる．

(12) 送信完了割り込みを通知

ソケットバッファを 1 つ送信し終えると，NIC は OS に向けて送信完了割り込みを通知する．その後．OS は割り込みに対応した処理をおこなう．送信完了割り込みは，1 パケット送信完了を契機として通知される．

(13) 未送信データの有無を確認

NIC は 1 パケット単位で送信処理を実行する．NIC ドライバは，NIC が送信処理を行っている間に，新たにパケット送信要求を通知してくる場合がある．このため，NIC は 1 パケット送信後に，未送信パケットの有無を確認する必要がある．NIC が未送信パケットを確認する様子を図 11 に示し，以下で説明する．

NIC の送信処理は 1 パケット単位で行われる．図の (1)，(2) で，NIC は送信ディスクリプタ 0，送信ディスクリプタ 1 を参照しパケット A を送信する．その後，図の (3) で次の送信ディスクリプタ 2 を参照する．この時，送信ディスクリプタ 2 が空の送信ディスクリプタだった場合，NIC は送信処理を終了する．図 11 で示すように，送信ディスクリプタ 2 が未送信パケットを持つ場合，NIC は再度パケット送信処理を実行する．

(14) 送信終了処理

未送信パケットをすべて送信し終えた後，NIC は送信処理を終了する．NIC は TPPoll を 0 にし，NIC ドライバからの送信要求を待つ．

7　おわりに

本資料では，パケット送信時の NIC の動作について調査した結果について述べた．

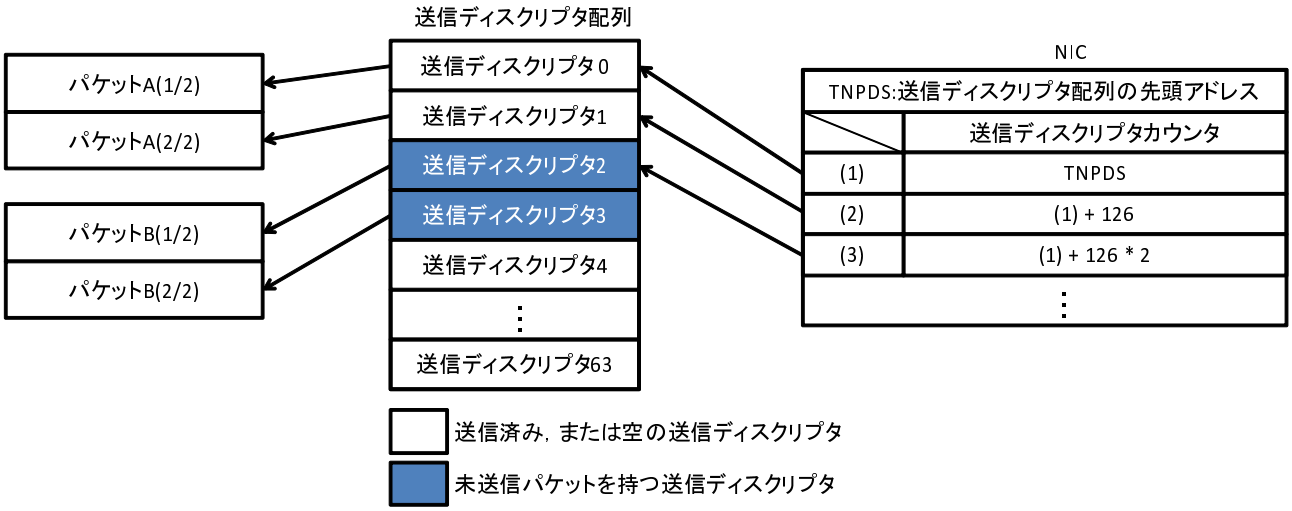

図 11　未送信パケットの有無を確認