NIC ドライバの呼び出し経路の調査

2013/10/15

増田陽介

1 はじめに

MSI のルーティング変更による NIC 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．

NIC を他 OS ノードに移譲した際の，移譲元 OS ノードのパケット送受信処理をフックしたいという要求がある．

パケット送受信処理は NIC ドライバ内の特定の関数で処理される．パケット送受信時の処理をフックするために，NIC ドライバの関数がどのような経路で呼び出されているかを調査した．

パケット送受信処理を行う 4 つの関数, rtl8169_interrupt(),rtl8169_tx_interrupt(), rtl8169_rx_interrupt(), および rtl8169_start_xmit() について，それぞれの関数がどのような経路で呼び出されているのかを調査した．

調査した結果，rtl8169_tx_interrupt() と rtl8169_rx_interrupt() は rt8169_poll() という関数ないでポーリング処理として実行されていることが分かった．

NIC ドライバへの IO をフックするために，改修が必要な箇所がわかった．rtl8169_poll() は net_rx_action() という関数で呼ばれている．また，rtl8169_start_xmit() は dev_queue_xmit() という関数で呼ばれている．rtl8169_interrupt() は do_IRQ() で呼ばれている．

2 呼び出し順序を調査した関数

パケット送受信次の処理流れを調査するために，NIC ドライバ内の関数の呼び出し経路を調査した．

呼び出し経路を調査した 4 つの関数の名前と機能の概要を以下に示す．

(1) rtl8169_start_xmit()
パケット送信処理時に呼び出される関数．DMA 領域に送信データを配置し，NIC にパケット送信を依頼する．

(2) rtl8169_tx_interrupt()
NIC からの割り込みを受けて，割り込みハンドラから呼び出される関数．パケット送信時に使用したバッファや構造体を初期化する．

(3) rtl8169_rx_interrupt()
NIC からの割り込みを受けて，割り込みハンドラから呼び出される関数．NIC が受信したデータをバッファから取り出し，受信に使用したバッファや構造体を初期化する．

(4) rtl8169_interrupt()
NIC からの割り込みを受けて，割り込みハンドラから呼び出される関数．NIC の状態を確認す

る．障害が起こっていれば，対応する．

3 呼び出し経路について

3.1 rtl8169_start_xmit()

rtl8169_start_xmit() の呼び出し経路について説明する．rtl8169_start_xmit() はパケット送信処理を行う関数である．割り込みを契機として開始する処理と異なり，パケット送信処理は処理を開始する契機が一意に特定できない．

そこで，本資料ではパケット送信処理の開始を以下のように定義する．まず，Linux ネットワークスタックのアーキテクチャを図に示す．

図の内ネットワーク・プロトコル以上の層では，送信時に必要な情報をソケットバッファに格納する処理が行われる．そして，デバイス非依存インタフェース以下で，プロトコル層から受け取ったソケットバッファをキューに格納し送信する処理が行われる．デバイス非依存インタフェースにおける処理のはじめに呼ばれる関数が dev_queue_xmit() である．本資料ではプロトコル層からソケットバッファを受け取る時点を，送信処理の開始と定義する．dev_queue_xmit() 以降の処理流れを図で示し，以下で説明する．

(1) dev_queue_xmit()
プロトコル層から渡されたソケットバッファを送信する関数．キューに送信バッファを積む処理を行う．ソケットバッファから送信デバイス情報を取得する．送信デバイス情報とは, net_device 構造体を指す．また，ソケットバッファから netdev_queue を取得する．netdev_queue には現在のネットデバイスのキュー情報が格納されている．Qdisc の enque に関数ポインタがセットされていれば，__dev_xmit_skb を実行.

(2) __dev_xmit_skb()
ソケットバッファを送信する処理を，中継する関数．ソケットバッファに格納されているデータを送信するために.sch_direct_xmit() を実行する．送信後，キュー内に未送信のソケットバッファが残っているならば__qdisc_run() を実行する．キュー内にデータが残っていなければ disc_run_end() を実行する．返り値としてネットワークの状態を返す．送信成功なら TX_OK, 失敗なら TX_BUSSY を返す.

(3) sch_direct_xmit()
キュー内のソケットバッファを 1 つ送信する関数．送信キューが使用可能なら，dev_hard_start_xmit() を呼び出す．送信に成功したなら，Qdisc 内の関数ポインタ qlean を実行．送信に失敗なら，再度キューにデータを入れなおす.

(4) dev_hard_start_xmit()
NIC を登録する際に設定された，NIC ドライバの送信処理関数を実行する関数．送信処理を行う関数は，net_device_ops 構造体中の関数ポインタ.ndo_start_xmit 登録されている.

(5) rtl8169_start_xmit()

NIC ドライバの送信処理を行う関数．DMA 領域にソケットバッファ内のデータをマッピングする．その後，MMIO を用いて NIC にパケット送信を依頼する．

3.2 rtl8169_tx_interrupt()

rtl8169_tx_interrupt() は，rtl8169_start_xmit() で使用した送信資源表の初期化を行う関数である．NIC からの割り込みを契機として呼び出される．

rtl8169_tx_interrupt() が呼び出される経路を図で示し，以下で図中の関数について説明する．

(1) ENTRY(interrupt) 割り込みのエントリ部分．行う処理はない．

(2) ENTRY(irt_entries_start) スタックに IRQ 番号を詰み，common_interrupt にジャンプする．詳細は不明．

(3) commmon_interrupt()
割り込み処理を中継するである．do_IRQ() を呼び出す．詳細は不明．

(4) do_IRQ()
デバイスからの割り込みを処理する関数である．common_interrupt() から呼び出される．スタックに積まれたベクタ番号を取り出し，ベクタ管理表から対応する IRQ 番号を導く．そして，この IRQ に対応する irq_desc 構造体の handle_irq 関数を実行する．

(5) irq_exit()
ハード割り込みの終了処理を行う関数．コンテクストスイッチで退避された情報をもとに戻す．local_softirq_pending() で現在実行している割り込みにソフト割り込みがあるかを確認する．ソフト割り込みがある場合，ソフト割り込み処理を開始するために invoke_softirq() を呼び出す．

(6) invoke_softirq()
ソフト割り込みを呼び出す関数．ハード割り込みの後に，ソフト割り込みを呼び出す場合とソフトウェアからソフト割り込みを呼び出している場合で処理が分岐する．割り込み処理の場合は do_softirq() を呼び出し，ソフトウェア割り込みの場合は wakeup_softirq() を呼び出す．

(7) do_softirq()
割り込みの受付を禁止する．local_softirq_pending() でソフト割り込みがあるか否かを確認する．ソフト関数がある場合，call_softirq() を呼び出す．ソフト割り込み処理終了後，割り込みを許可する．

(8) Entry(call_softirq)
スタックに積んであるソフト割り込みを呼ぶ処理__do_softirq() を呼び出す．

(9) __do_softirq()
割り込みを許可する．これによりソフト割り込み処理中に割り込みが発生する．ソフト割り込み処理のテーブル softirq_irq を h に設定する．関数ポインタ h-¿action(h) を順番に呼び出す．h-¿action(h) が指すのはソフト割り込みハンドラである．すべてのソフト割り込みを処理するまでループする．ループ終了後，割り込みを使用不能にする，

(10) net_rx_action()
パケット受信処理を実行するソフト割り込みハンドラ．割り込みを禁止する．napi_struct 内の関数ポインタ poll を呼び出し，NIC が受信したパケットを取り出すポーリング処理を行う．バッファが空になるとポーリング処理を終了．割り込みを許可する．

(11) rtl8169_poll()
RTL8169 のポーリング処理を処理する関数．rtl8169_tx_interrupt() と rtl8169_rx_interrupt() を呼び出す．

(12) rtl8169_tx_interrupt() rtl8169_start_xmit() で使用した送信管理表を初期化する．

3.3 rtl8169_rx_interrupt()

rtl8169_rx_interrupt() は NIC が受信したパケットを受け取る関数である．NIC は受信したパケットを受信管理表に格納する．NIC ドライバは受信管理表に割り当てられているバッファから，受信パケットを取り出す．その後，パケットを取り出した受信管理表を初期化する．

rtl8169_rx_interrupt() が呼び出される過程を以下に図で示し，図中に登場する関数について説明する．

(1) ENTRY(interrupt)
割り込みのエントリ．

(2) ENTRY(irt_entries_start)
スタックに IRQ 番号を詰み，common_interrupt をコールする処理．詳細は不明．

(3) commmon_interrupt()
割り込み処理を中継するである．do_IRQ() を呼び出す．詳細は不明．

(4) do_IRQ()
デバイスからの割り込みを処理する関数である．common_interrupt() から呼び出される．スタックに積まれたベクタ番号を取り出し，ベクタ管理表から対応する IRQ 番号を導く．そして，この IRQ に対応する irq_desc 構造体の handle_irq 関数を実行する．

(5) irq_exit()
ハード割り込みの終了処理を行う関数．コンテクストスイッチで退避された情報をもとに戻す．local_softirq_pending() で現在実行している割り込みにソフト割り込みがあるかを確認する．ソフト割り込みがある場合，ソフト割り込み処理を開始するために invoke_softirq() を呼び出す．

(6) invoke_softirq()
ソフト割り込みを呼び出す関数．ハード割り込みの後に，ソフト割り込みを呼び出す場合とソフトウェアからソフト割り込みを呼び出している場合で処理が分岐する．割り込み処理の場合は do_softirq() を呼び出し，ソフトウェア割り込みの場合は wakeup_softirq() を呼び出す．

(7) do_softirq()
割り込みの受付を禁止する．local_softirq_pending() でソフト割り込みがあるか否かを確認す

る．ソフト関数がある場合，call_softirq() を呼び出す．ソフト割り込み処理終了後，割り込みを許可する．

(8) Entry(call_softirq)
スタックに積んであるソフト割り込みを呼ぶ処理__do_softirq() を呼び出す．

(9) __do_softirq()
割り込みを許可する．これによりソフト割り込み処理中に割り込みが発生する．ソフト割り込み処理のテーブル softirq_irq を h に設定する．関数ポインタ h-¿action(h) を順番に呼び出す．h-¿action(h) が指すのはソフト割り込みハンドラである．すべてのソフト割り込みを処理するまでループする．ループ終了後，割り込みを使用不能にする，

(10) net_rx_action()
パケット受信処理を実行するソフト割り込みハンドラ．割り込みを禁止する．napi_struct 内の関数ポインタ poll を呼び出し，NIC が受信したパケットを取り出すポーリング処理を行う．バッファが空になるとポーリング処理を終了．割り込みを許可する．

(11) rtl8169_poll()
RTL8169 のポーリング処理を処理する関数．rtl8169_tx_interrupt() と rtl8169_rx_interrupt() を呼び出す．

(12) rtl8169_tx_interrupt() rtl8169_start_xmit() で使用した送信管理表を初期化する．

(13) rtl8169_rx_interrupt() 受信後の処理を行う関数．使用済みの受信管理表からパケットを取り出す．パケットをソケットバッファに格納して上位層に挙げる．

3.4 rtl8169_interrupt()

rtl8169_interrupt() は，NIC の割り込みに対するハンドラとして登録されている関数である．NIC のレジスタを参照し，NIC やコネクションの状態を確認している．

rtl8169_interrupt() が呼び出される過程を以下で図示し，図中に登場する関数について説明する．

(1) common_interrupt
割り込み処理を中継するである

(2) do_IRQ()
デバイスからの割り込みを処理する関数である．common_interrupt() から呼び出される．スタックに積まれ t ベクタ番号を取り出し，ベクタ管理表から対応する IRQ 番号を導く．そして，この IRQ に対応する irq_desc 構造体の handle_irq 関数を実行する．

(3) handle_irq()
スタックがオーバフローしていないかをチェック irq 番号からしようする irq_desc 構造体を取得．その後，generic_handle_irq() を呼び出す．

(4) generic_handle_irq_desc()
中継する関数．irq_desc 構造体内の関数ポインタ handle_irq を用いて割り込みハンドラを呼び

出す．割り込みがレベルトリガ方式ならば，handle_level_irq() を呼び出す．割り込みがエッジトリガ方式ならば，handle_edge_irq() を呼び出す．MSI はエッジトリガ方式の割り込みなので，handl_edge_irq() を呼び出す．

(5) handle_edge_irq()
エッジトリガ方式の割り込みを処理するための関数．割り込み処理中ならマスクを掛ける．irq_desc 構造体を handle_irq_event() に渡す．

(6) handle_irq_event()
irq_desc に割り込み処理中であることを設定．スピンロックを解除して，割り込みハンドラ起動関数である handle_irq_event_percpu を呼び出す．割り込み処理終了後，irq_desc の割り込み中設定を解除．

(7) handle_irq_event_percpu()
irqaction 構造体中の関数ポインタである handeler から，割り込みハンドラを呼び出す．irqaction はリスト構造になっており，handle_irq_event_percpu はリスト中のすべての handler を順に呼び出す．

(8) rtl8169_interrupt()
RTL8169 のハード割り込みハンドラ．NIC に障害が発生していないか確認する．

4 ポーリング処理

NIC ドライバの呼び出し経路について調査したところ, rtl8169_tx_interrupt() と rtl8169_rx_interrupt() が，rtl8169_poll() という関数内で呼ばれていることがわかった．2 つの関数が rtl8169_poll() から呼ばれているのは，NIC ドライバが割り込みに対してポーリング処理を行っているためである．NIC ドライバが行うポーリング処理について説明する．

NIC ドライバは NAPI(New API) と呼ばれる機構を採用している．NAPI は割り込み処理の負荷を低減するための機構である．従来の受信処理ではパケット受信のたびにハードウェア割り込みを実行していたため，パケットを大量に受信した際割り込み処理による負荷が増大してしまう問題があった．

この問題を解決するために，NAPI では一度パケットを受信してハードウェア割り込みが発生すると，ハードウェア割り込みを禁止して，ポーリング処理によって受信パケットを処理する．バッファが空になるか一定回数ポーリング処理を実行すると，ポーリング処理を停止しハードウェア割り込みを受け入れるようにする，

rtl8169_poll() は, 受信割り込みのハンドラである net_rx_interrupt() で呼び出されている．RTL8169 における，パケット受診時のポーリング処理を以下に図で示し説明する．

5 NIC の IO をフックするために改変を加える必要がある関数

net_rx_action とで dev_hard_xmit を改変する．

6　問題点と解決案

前章で NIC ドライバがポーリング処理を行っていることを示した．発生すると思われる問題を以下に示す．

(1) NIC 移譲後にポーリング処理を続行送信途中で移譲された場合，

7　MSI ルーティング変更による NIC の移譲に関する考察

調査結果から NIC への入出力要求の際に通る関数が分かった．NIC を移譲した際，これらの関数を変更することで NIC を使用できなくすることができると考えれる．

8　おわりに

関数の呼び出し順序を調査した．調査結果から，移譲時に改修を加えるべき関数が判明した．