

12. Realtek 8139 Ethernet Controller

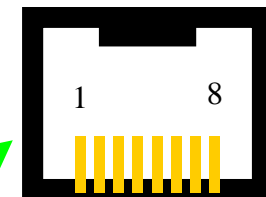
12.1.1 Ethernetvarianten

- 10 BASE T: 10MBit Manchester Codierung.
- 100 BASE T/TX: 100MBit 4B/5B Codierung:
 - full-duplex mit passendem Switching-Hub, oder halb-duplex.

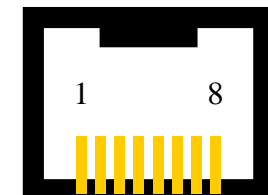
12.1.2 RJ-45 Steckerformat

- Stiftbelegung:
 - normalerweise keine Überkreuzung im Kabel,
 - 8 Stifte, davon nur 4 Stifte benutzt,
 - an Steckdosen & Downlink an Hubs und Switches
 - an Netzwerkkarte & Uplink am Hub oder Switch

1 Transmit +
2 Transmit -
3 Receive +
6 Receive -



1 Receive -
2 Receive +
3 Transmit +
6 Transmit -



12.2

Chip-Architektur des 8139

- PCI-Schnittstelle:

- Bus Master Übertragung,
- DMA Deskriptoren.

- Interruptsteuerung:

- Senden, empfangen,
- Fehlersituationen.

- Zusätzliche Features:

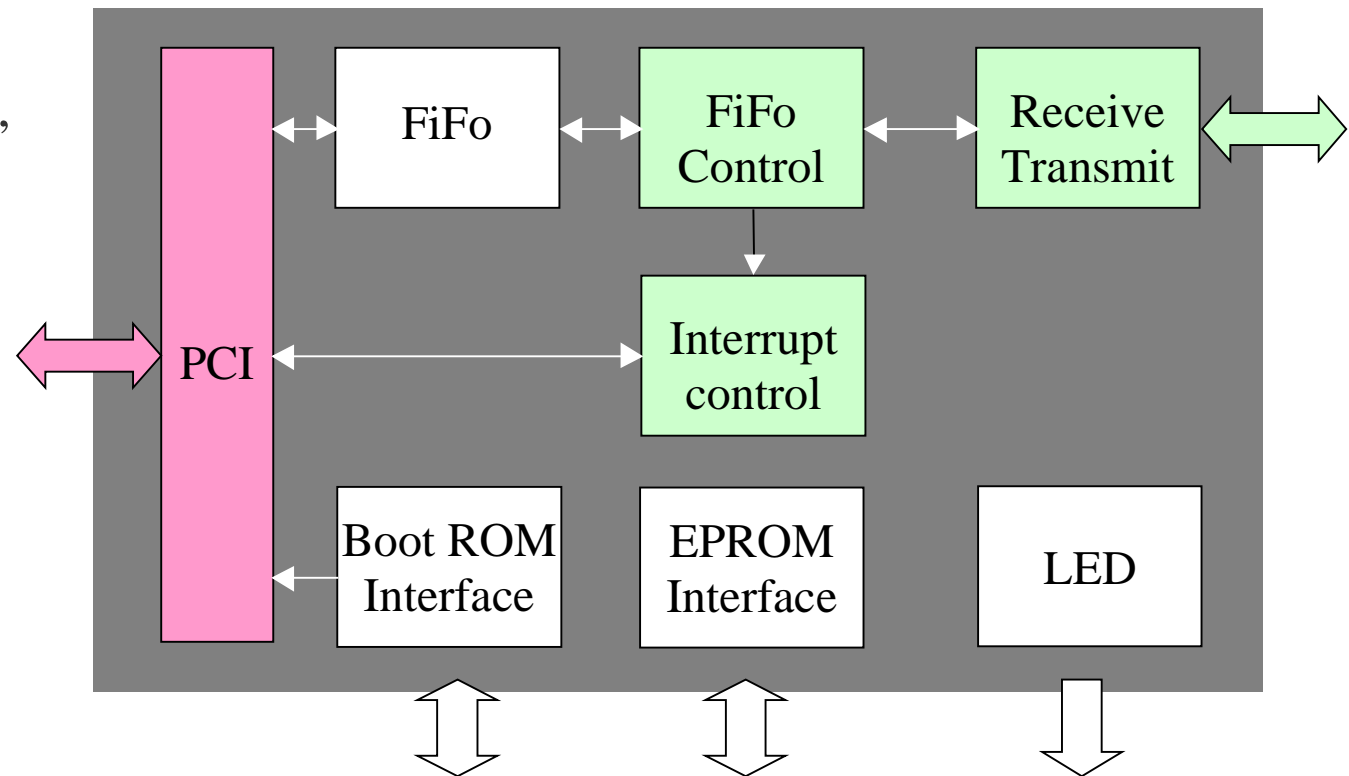
- Auto-Negotiation
- Power-Control,
- LAN Wakeup,
- Flow Control.

- Boot ROM zum Starten vom Netz.

- LED zur Anzeige des Link-Zustandes.

- EEPROM zur Ablage von Konfigurierungsparametern.

- FIFO Steuerung mit Schwellwert-Einstellungen ...



12.3 Arbeitsweise

- Transfer nur über Bus Mastering möglich:

- Ältere Boards bieten nicht auf allen Sockeln Bus Mastering,
- RTL 8139 arbeitet auf physikalischen Adressen.

- Pufferung geschieht im Hauptspeicher:

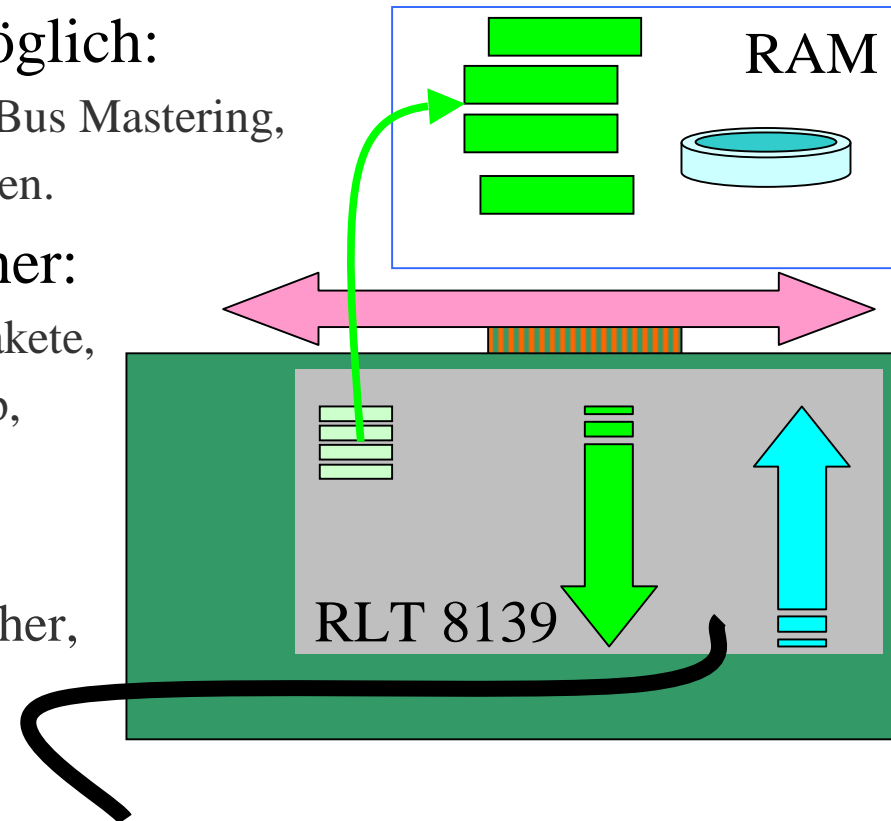
- 64 KByte Ringpuffer für die empfangenen Pakete,
- 4 Deskriptoren für abgehende Pakete im Chip,
- Pakete selber im Hauptspeicher.

- 4 Transmit-Deskriptoren:

- Adresse des Sendepuffers im Hauptspeicher,
- Zustand der Übertragung,
- Länge des Paketes.

- Zwei FiFo Puffer à je 2Kbyte auf dem Chip (Senden & Empfangen):

- beim Erreichen eines Schwellwertes im E.Ppuffer beginnt DMA Übertragung,
- beim Erreichen eines Schwellwertes im Sendepuffer beginnt Übertragung ins Netz.



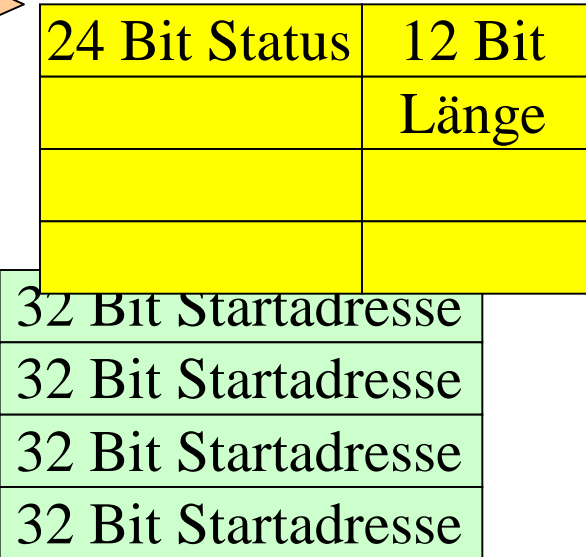
12.3.1 Transmit-Deskriptoren

• Transmit Status Register - TSD0 .. TSD3 (32 Bit, \$10):

- [31] Carrier Sense Lost
- [30] Transmit Abort
- [29] Out of Window Collision
- [28] CD Heart Beat (nur bei 10Mb)
- [27..24] Carrier Sense Lost
- [21..16] Early Transmit Threshold:
 - 8 ... n*32 Bytes,
- [15] TOK, Transmit successful,
- [14] TUN, Transmit FiFo underrun,
- [13] OWN,
 - the CPU currently **owns** the packet
 - not the adapter.
- [12..0] Größe des zu versendenden Paketes in Bytes

TSD0 @ \$10

TSAD0 @ \$20

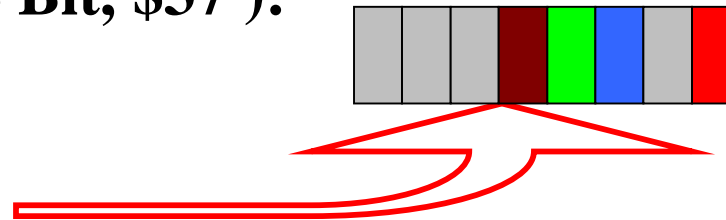


• Transmit Start Adress Register - TSAD0 .. TSAD (32 Bit, \$20).

12.3.2 Weitere Gerätereister

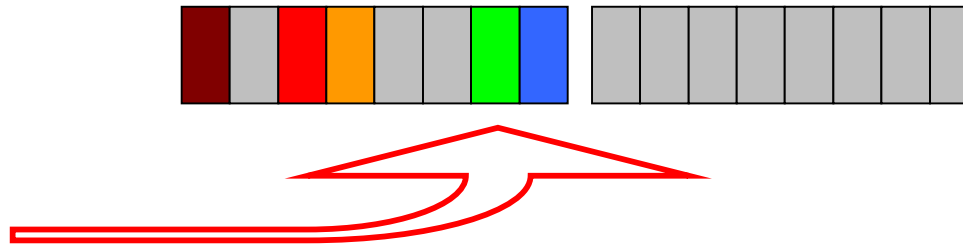
- ca. 60 Register vorhanden, Memory mapped.
- **CR - Command Register (8 Bit, \$37):**

- [4] **Reset**
- [3] **Receiver Enable**
- [2] **Transmitter Enable**
- [0] **Buffer empty**



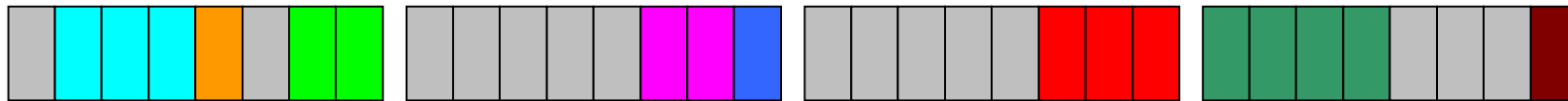
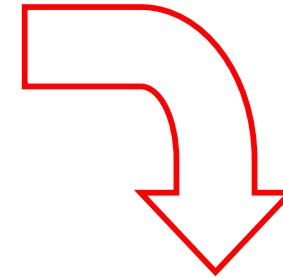
- **BMCR - Basic Mode Control Register (16 Bit, \$62):**

- [15] **Reset**
- [13] **Speed_Set 100Mbps**
- [12] **Auto Negotiation enabled**
- [9] **Restart Auto Negotiation**
- [8] **Enable full-duplex mode**



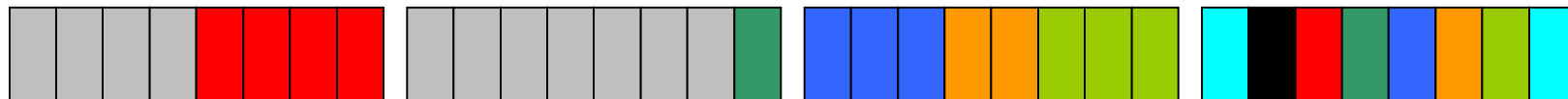
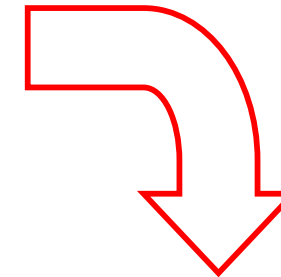
• TCR - Transmit Configuration Register (32 Bit, \$40)

- [30..28] Hardware Versionsnummer
- [27] RTL8139B(L)
- [25..24] Interframe Gap 9.6us .. 8.4us (100Mbit) oder 960ns .. 840 ns (10Mbit)
- [18..17] (normal, MAC, PHY, Twister)-Loopback,
- [16] CRC Prüfsumme am Paketende anfügen
- [10..8] Max Tx-DMA Burst 16-2048 Bytes
- [7..4] TxRetryCount: Retries=16 + (Tx Retry*16)
- [0] Clear Abort



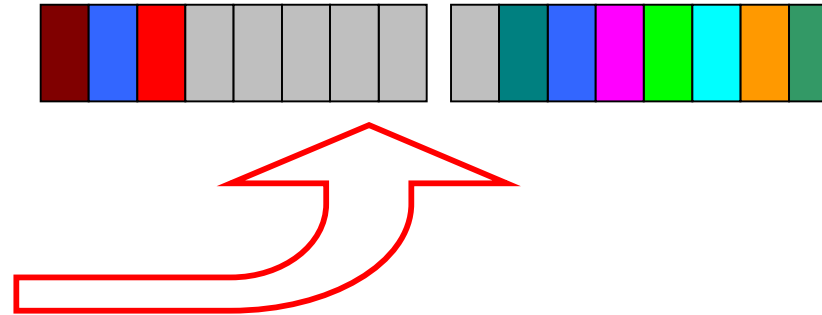
• RCR - Receive Configuration Register (32 Bit, \$44):

- [27..24] **Early Receive Threshold zum Auslösen des Interrupts (% der Gesamtlänge):**
 0000 = no early threshold, 0001 = 1/16, 0010 = 2/16 1111 = 15/16
 Nur möglich für bekannte Protokolle mit Längensfeld.
- [16] fehlerhafte Pakete mit Länge > 8 Bytes empfangen
- [15..13] Schwellwert für Empfangs-DMA (16 Bytes ... 1024 Bytes, 111 = ganzes Paket)
- [12..11] Länge des Empfangspuffers (00 = 8K ; 01 =16K ... 11 = 64K)
- [10..8] Maximale DMA Burst Size per Rx DMA (16 Bytes .. unbegrenzt)
- [7] sofortiger Wraparound am Ende des Ringpuffers oder erst nach einem Paket
- [6] Eprom Typ (nicht alle Chipversionen)
- [5] Fehlerhafte Pakete annehmen
- [4] Pakete < 64 Bytes annehmen
- [3] Broadcast Pakete annehmen
- [2] Multicast Pakete annehmen,
- [1] Pakete mit übereinstimmender MAC- Adresse annehmen
- [0] Pakete mit beliebiger MAC Adresse annehmen



Interrupt Status & Interrupt Mask Register (16 Bit, \$3e / \$3c):

- [15] System Error
- [14] Time Out
- [13] Cable Length changed
- [6] Receive Fifo Overflow
- [5] Link changed
- [4] Rx Buffer overflow
- [3] Transmit Error Interrupt
- [2] Transmit OK Interrupt
- [1] Receive Error Interrupt
- [0] Receive OK Interrupt



- **RBSTART- Receive Buffer Start Address (32 Bit, \$30):**
 - Offset an der das nächste zu lesende Paket steht,
 - maximal 64 Kbyte Puffer möglich.
- **CAPR - Current Address of Packet Read (16 Bit, \$38):**
 - Offset an der das nächste zu lesende Paket steht,
 - maximal 64 Kbyte Puffer möglich,
 - aktueller Lesezeiger der Karte.
- **CBA - Current Buffer Address (16 Bit, \$3a):**
 - aktueller Schreibzeiger der Karte,
 - Wrap-Around am Ende des Puffers.
- **IDR0..IDR5 - Mac ID Register (48 Bit, \$00):**
 - Ethernet-Adresse der Karte,
 - gegebenenfalls selber setzen.
- **MAR0..MAR7 - Multicast Address (64 Bit, \$08):**
 - Bitmap für den Hashcode der Multicast-Adresse,
 - Detailtest durch die Software.

12.4 Registerklasse für RLT 8139

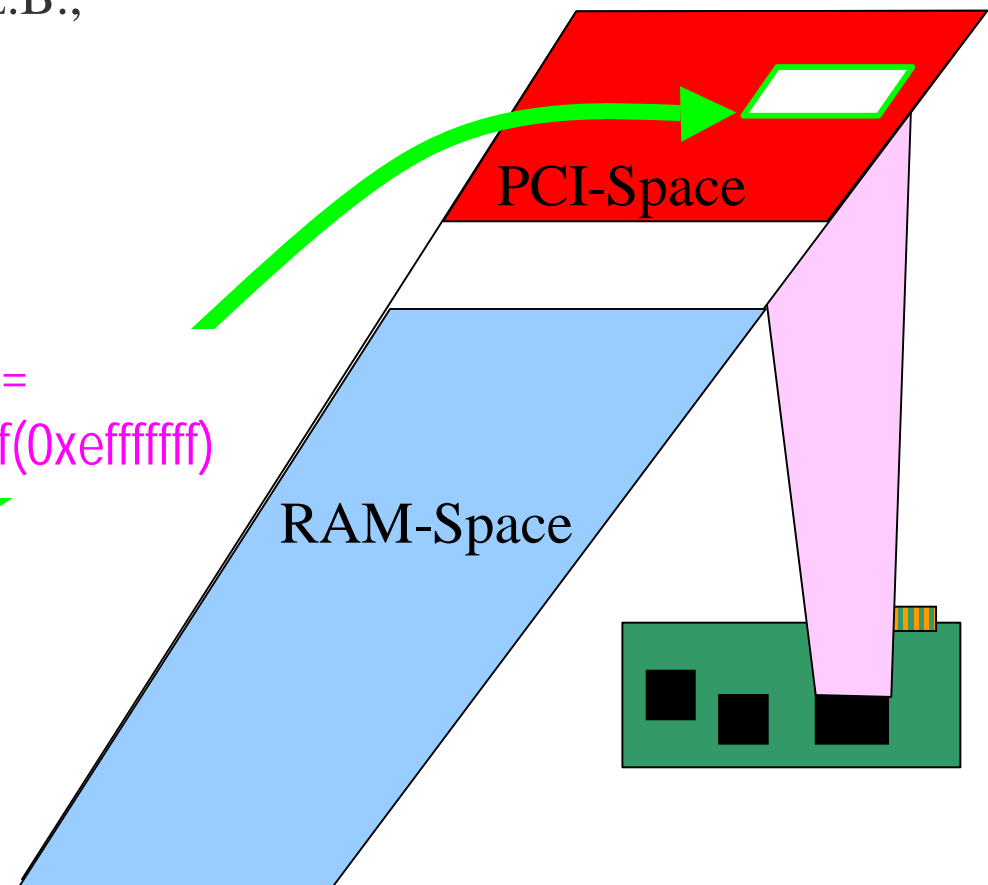
12.4.1 Pseudoklasse STRUCT in Plurix:

- Basisklasse für memory-mapped Geräte-Register,
 - bildet Memory-Mapped Gerätereister auf den Hauptspeicher ab,
 - für Grafik-, Sound- und Netzwerkkarten z.B.,
 - kann nicht mit new() alloziert werden,
 - z.B. :

```
public class Regs8139 extends STRUCT
```

```
SYSTEM.nicDriver.registers =  
(Regs8139)MAGIC.Cast2Ref(0xefffffff)
```

Regs8139



- Referenzierung:

- Normalerweise über eine besondere Referenzvariable im zugehörigen Gerätetreiber,
- Referenz wird mit MAGIC.Cast(„myType“, Hardware-Adresse) gebildet,
- In Sprachen wie C, Pascal, Modula-2 als Struct bzw. Record,
- In Plurix als Compiler-Feature implementiert.

- Pseudoklasse mit „Hardware“-Variablennamen:

- Mnemonischer und effizienter als Methodenaufrufe,
- Kann keine normalen Programmvariablen besitzen.

```
package devices;
/** Bildet die RLT8139 Register in den PCI-Speicher ab
 * @author P. Schulthess
 * @version 0.001
 */

public class Regs8139 extends STRUCT {
    int IDR0, IDR4, MAR0, MAR4;           // 6 Byte MAC-Address, 2 B Filler, 8 B Multicast Mask
    int TSD0, TSD1, TSD2, TSD3;         // Transmit status of descriptors 0..3, $10 ...
    int TSAD0, TSAD1, TSAD2, TSAD3;     // Transmit address of descriptors 0..3, $20 ..
    int RBSTART;                         // receive buffer start address, $30 .
    short ERBCR;                         // early receive byte count, $34
}
```

```

byte ERSR, CR; // early receive status, command register, $36
short CAPR, CBR; // curr. adr. of pk read, curr. buffer adr.
short IMR, ISR; // Int mask, Int status
int TCR, RCR; // Transmit/Receive configuration register,
int TCTR, MPC; // timer count, missed packets, $40
byte CR9346, CONFIG0, CONFIG1, f1; // 9346 command, configuration1/2,, $50
int TimerInt; // Timer interval until timeout interupt in ISR
byte MSR, CONFIG3, f2, f3; // Media status, configuration 3,, $58
short MULINT, RERID, TSAD; // Multiple Int, PCI Revision, xMit status all
short BMCR, BMSR; // Basic mode control/-status,
short ANAR, ANLPAR, ANER; // autoneg. advertise/-partner/-expansion, $$66
short DIS, FSC; // disconnect counter, false carrier
short NWAYTR, RX_ER, CSCR, f4; // N-wy test, receive errors, cs configuration, $70
long PHY1_P, TW_P, PHY2_P; // Physical parameters, twist p., 3 bytes filler, $78
byte CRC0, CRC4; // Power management and wakeup, $84
long W0, W1, W2, W3; // wakeup frames, $8c
long W4, W5, W6, W7; // wakeup frames
byte LSB0, LSB4; // Power management and wakeup, $cc

```

```

public void initRegisters( byte[] rxBuffer) {
    CR= (byte) 0x10;      // reset adapter
    while (CR >(byte) 0); // reset pending
    CR9346=(byte) 0xC0;  // enable Config_Register_Write
    ANAR= (short) 0x3E1; // all modes advertised, CSMA#1
    BMCR= (short) 0x2000; // no auto negotiation, 100 Mbps
    IMR= (short) 0;      // disable all NIC interrupts
    RCR=(byte)          // receive command register
        ( 3 <<11)      // RxBufferSize: 0=8K, 1=16K, 2=32K, 3=64K
        |( 7<<1)       // RxFifoThreshold: 0=16Bytes, 1=32Bytes, .. 7=no rx threshold
        |( 6 <<8);      // MaxRxDmaBurst: 0=16Bytes, 1=32Bytes, .. 7=unlimited
    TCR=(byte)          // transmit command register
        ( 6 <<8)       // MaxTxDmaBurst: 0=16Bytes, 1=32Bytes, .. 7=2048bytes
        |( 3 <<24);     // InterframeGapTime: 9.6us 10MBit, 960ns 100MBit
    CR9346=(byte) 0;    // disable Config_Register_Write
    RBSTART=(Regs8139)MAGIC.Cast2Ref(rxBuffer); // Startaddress of receive buffer
}
}

```

- Aufruf z.B. mit:

```
SYSTEM.nicDriver.registers.init( packetBuffer );
```