Posted by wolverin on Tue, 01 Aug 2023 18:07:03 GMT
View Forum Message <> Reply to Message

https:// learn.microsoft.com/ru-ru/windows/win32/api/iptypes/ns-iptyp
es-ip_adapter_addresses_lh?redirectedfrom=MSDN

```
const
  DLL = 'IPHLPAPI.DLL';

  MAX_ADAPTER_ADDRESS_LENGTH = 8;
  MAX_DHCPV6_DUID_LENGTH = 130;

type
  SOCKET_ADDRESS = record end;

  PIP_ADAPTER_UNICAST_ADDRESS = ^IP_ADAPTER_UNICAST_ADDRESS;
  IP_ADAPTER_UNICAST_ADDRESS = record union: record
    case Integer of
      0: (Alignment: UInt64);
      1: (Length: ULONG;
          Flags: DWORD);
    end;
    Next: PIP_ADAPTER_UNICAST_ADDRESS;
    Address: SOCKET_ADDRESS;
    //PrefixOrigin: IP_PREFIX_ORIGIN;
    //SuffixOrigin: IP_SUFFIX_ORIGIN;
    //DadState: IP_DAD_STATE;
    ValidLifetime: ULONG;
    PreferredLifetime: ULONG;
    LeaseLifetime: ULONG;
    OnLinkPrefixLength: BYTE;
    end;

  PIP_ADAPTER_ANYCAST_ADDRESS = ^IP_ADAPTER_ANYCAST_ADDRESS;
  IP_ADAPTER_ANYCAST_ADDRESS = record union: record
    case Integer of
      0: (Alignment: UInt64);
      1: (Length: ULONG;
          Flags: DWORD);
    end;
    Next: PIP_ADAPTER_ANYCAST_ADDRESS;
    Address: SOCKET_ADDRESS;
```

```
    end;

  PIP_ADAPTER_MULTICAST_ADDRESS = ^IP_ADAPTER_MULTICAST_ADDRESS;
  IP_ADAPTER_MULTICAST_ADDRESS = record end;

  PIP_ADAPTER_DNS_SERVER_ADDRESS = ^IP_ADAPTER_DNS_SERVER_ADDRESS;
  IP_ADAPTER_DNS_SERVER_ADDRESS = record end;

  PIP_ADAPTER_ADDRESSES = ^IP_ADAPTER_ADDRESSES;
  IP_ADAPTER_ADDRESSES = record union: record
    case Integer of
      0: (Alignment: UInt64);
      1: (Length: ULONG;
          IfIndex: DWORD);//IF_INDEX IfIndex;
    end;
    Next: PIP_ADAPTER_ADDRESSES;
    AdapterName: PCHAR;
    FirstUnicastAddress: PIP_ADAPTER_UNICAST_ADDRESS;
    FirstAnycastAddress: PIP_ADAPTER_ANYCAST_ADDRESS;
    FirstMulticastAddress: PIP_ADAPTER_MULTICAST_ADDRESS;
    FirstDnsServerAddress: PIP_ADAPTER_DNS_SERVER_ADDRESS;
    DnsSuffix: PWCHAR;
    Description: PWCHAR;
    FriendlyName: PWCHAR;
    PhysicalAddress: array[1..MAX_ADAPTER_ADDRESS_LENGTH] of BYTE;
    PhysicalAddressLength: ULONG;
//    union: record
{    case Integer of
      0: (Flags: ULONG);
      1: (DdnsEnabled: ULONG = 1;
          RegisterAdapterSuffix: ULONG = 1;
          Dhcpv4Enabled: ULONG = 1;
          ReceiveOnly: ULONG = 1;
          NoMulticast: ULONG = 1;
          Ipv6OtherStatefulConfig: ULONG = 1;
          NetbiosOverTcpipEnabled: ULONG = 1;
          Ipv4Enabled: ULONG = 1;
          Ipv6Enabled: ULONG = 1;
          Ipv6ManagedAddressConfigurationSupported: ULONG = 1;
    end;}
    Mtu: ULONG;
    IfType: DWORD;//IFTYPE
    //OperStatus: IF_OPER_STATUS;
    Ipv6IfIndex: DWORD;//IF_INDEX
    ZoneIndices: array[0..15] of ULONG;
    //FirstPrefix: PIP_ADAPTER_PREFIX;
    TransmitLinkSpeed: UInt64;
    ReceiveLinkSpeed: UInt64;
```

```
    //FirstWinsServerAddress: PIP_ADAPTER_WINS_SERVER_ADDRESS;
    //FirstGatewayAddress: PIP_ADAPTER_GATEWAY_ADDRESS;
    Ipv4Metric: ULONG;
    Ipv6Metric: ULONG;
    //Luid: IF_LUID;
    Dhcpv4Server: SOCKET_ADDRESS;
    //CompartmentId: NET_IF_COMPARTMENT_ID;
    //NetworkGuid: NET_IF_NETWORK_GUID;
    //ConnectionType: NET_IF_CONNECTION_TYPE;
    //TunnelType: TUNNEL_TYPE;
    Dhcpv6Server: SOCKET_ADDRESS;
    Dhcpv6ClientDuid: array[1..MAX_DHCPV6_DUID_LENGTH] of BYTE;
    Dhcpv6ClientDuidLength: ULONG;
    Dhcpv6Iaid: ULONG;
    //FirstDnsSuffix: PIP_ADAPTER_DNS_SUFFIX;
  end;




 union {
   ULONG Flags;
   struct {
     ULONG DdnsEnabled : 1;
     ULONG RegisterAdapterSuffix : 1;
     ULONG Dhcpv4Enabled : 1;
     ULONG ReceiveOnly : 1;
     ULONG NoMulticast : 1;
     ULONG Ipv6OtherStatefulConfig : 1;
     ULONG NetbiosOverTcpipEnabled : 1;
     ULONG Ipv4Enabled : 1;
     ULONG Ipv6Enabled : 1;
     ULONG Ipv6ManagedAddressConfigurationSupported : 1;
   };
 };
```

Posted by on Wed, 02 Aug 2023 09:24:36 GMT
View Forum Message <> Reply to Message

Posted by wolverin on Wed, 02 Aug 2023 10:18:40 GMT
View Forum Message <> Reply to Message

Posted by on Wed, 02 Aug 2023 10:34:31 GMT
View Forum Message <> Reply to Message

Posted by SD on Wed, 02 Aug 2023 11:40:13 GMT
View Forum Message <> Reply to Message

Posted by skroik on Fri, 19 Jul 2024 13:46:53 GMT
View Forum Message <> Reply to Message

https://en.delphipraxis.net/topic/11864-union-with-bitfields /

C++-Code

```
// define QCL data types
typedef short   QCL_WORD;

typedef union {
   QCL_WORD data;
   struct {
      QCL_WORD alarm_state          :1;
            // bus flags: alarm state
```

```
        QCL_WORD alignment_state       :1;
              // bus flags: alignment state
        QCL_WORD inversion_wait        :1;
              // bus flags: external synchronism - inversion wait
        QCL_WORD out_of_min_bound      :1;
              // bus flags: column position is out of minimum boundaries
        QCL_WORD out_of_max_bound      :1;
              // bus flags: column position is out of maximum boundaries
        QCL_WORD below_safety_height   :1;              //
bus flags: column position is below safety height
        QCL_WORD in_working_pos        :1;
              // bus flags: column position is in working position
        QCL_WORD locking_latch_state   :1;              //
bus flags: column position in locking latch state
        QCL_WORD battery_warning       :1;
              // bus flags: column in battery warning
        QCL_WORD battery_alarm         :1;
              // bus flags: column in battery alarm
        QCL_WORD evr_state             :1;
              // bus flags: column EVR state
        QCL_WORD load_weight_zone      :2;
              // bus flags: column in load weight zone
        QCL_WORD column_is_consistent  :1;              //
bus flags: column in load weight zone
        QCL_WORD lift_set_acquire_req  :1;              // bus
flags: columns lift set acquire request
        QCL_WORD movement_mode_absolute :1;             //
bus flags: column has movement mode absolute
    } fields;
} t_bus_flags_w4;
```

Delphi-Code

```
type
  QCL_WORD = SmallInt;

  t_bus_flags_w4 = record
  private
    data: QCL_WORD;
    function GetBit(Index: Integer): Boolean;
    procedure SetBit(Index: Integer; Value: Boolean);
  public
    property AlarmState: Boolean index 0 read GetBit write SetBit;
    property AlignmentState: Boolean index 1 read GetBit write SetBit;
    property InversionWait: Boolean index 2 read GetBit write SetBit;
    property OutOfMinBound: Boolean index 3 read GetBit write SetBit;
    property OutOfMaxBound: Boolean index 4 read GetBit write SetBit;
```

```
    property BelowSafetyHeight: Boolean index 5 read GetBit write SetBit;
    property InWorkingPos: Boolean index 6 read GetBit write SetBit;
    property LockingLatchState: Boolean index 7 read GetBit write SetBit;
    property BatteryWarning: Boolean index 8 read GetBit write SetBit;
    property BatteryAlarm: Boolean index 9 read GetBit write SetBit;
    property EVRState: Boolean index 10 read GetBit write SetBit;
    property LoadWeightZone: Byte index 11 read GetBit write SetBit; // 2 Bits
    property ColumnIsConsistent: Boolean index 13 read GetBit write SetBit;
    property LiftSetAcquireReq: Boolean index 14 read GetBit write SetBit;
    property MovementModeAbsolute: Boolean index 15 read GetBit write SetBit;
  end;

implementation

function t_bus_flags_w4.GetBit(Index: Integer): Boolean;
begin
  Result := (data and (1 shl Index)) <> 0;
end;

procedure t_bus_flags_w4.SetBit(Index: Integer; Value: Boolean);
begin
  if Value then
    data := data or (1 shl Index)
  else
    data := data and not (1 shl Index);
end;

// For 2-Bit-Property LoadWeightZone we need some special Getter and Setter

function t_bus_flags_w4.GetLoadWeightZone: Byte;
begin
  Result := (data shr 11) and 3; // Extract 2 Bits at Position 11
end;

procedure t_bus_flags_w4.SetLoadWeightZone(Value: Byte);
begin
  data := (data and not ($3 shl 11)) or ((Value and $3) shl 11); // Set 2 Bits at Position 11
end;
```