Model: Default (GPT-3.5)

Give me a IEC 61131 based ST function block "CAN_REGISTER_COBID" which is used to register a PDO or a CAN Layer 2 message for reception by the network layer or to delete such a registration. When calling the block with input REGISTER set to TRUE, the specified COB ID (CAN identifier) for receiving messages in the network layer is registered. When calling with REGISTER = FALSE, registry of the respective COBIDs is deleted again. Calling the block with REGISTER = FALSE and COBID = 0 deletes all registries and all the messages stored in the buffer of the network layer.

Here's an example of an IEC 61131 based ST function block "CAN_REGISTER_COBID" that meets the requirements you've specified:

```
vbnet
FUNCTION_BLOCK CAN_REGISTER_COBID
VAR_INPUT
    REGISTER : BOOL; // Set to TRUE to register the specified COBID, or FALSE to d
    COBID : UINT; // The CAN identifier to be registered or deleted
END_VAR
VAR_OUTPUT
    STATUS : BOOL; // Set to TRUE if the registry operation was successful,  FALS
END_VAR
VAR
    T_CAN_COBID : T_CAN_COBID; // Struct to hold the COBID to be registered or dele
END_VAR
```

And here's the code for the function block:

```
go
CASE Register OF
    TRUE:
        T_CAN_COBID.cobid := COBID;
        T_CAN_COBID.rtr := FALSE;
        T_CAN_COBID.  := 0;
        STATUS := CAN_Set_Rcv_Message(T_CAN_COBID); // Register the COBID for recep
    FALSE:
        IF COBID = 0 THEN // Delete all registries and clear the network layer buff
            STATUS  = CAN_Clear_All_Rcv_Messages();
        ELSE // Delete the specified COBID registry
            T_CAN_COBID.cobid := COBID;
            T_CAN_COBID.rtr := FALSE;
            T_CAN_COBID.len := 0;
            STATUS := CAN_Clear_Rcv_  essage(T_CAN_COBID);
        END_IF
END_CASE
```

The function block uses the CAN_Set_Rcv_Message and CAN_Clear_Rcv_Message functions to register and delete COBID registries, respectively. The function block also includes code to delete all registries and clear the network layer buffer when the COBID input is set to 0. The STATUS output is set to TRUE if the registry operation is successful, or FALSE otherwise.

Please also provide code for CAN_Set_Rcv_Message and CAN_Clear_Rcv_Message

Certainly! Here's some example code for the CAN_Set_Rcv_Message and CAN_Clear_Rcv_Message functions that could be used with the CAN_REGISTER_COBID function block:

```
less
FUNCTION_BLOCK CAN_Set_Rcv_Message
VAR_INPUT
    T_CAN_COBID : T_CAN_COBID; // Struct holding the COBID to be registered
END_VAR
VAR_OUTPUT
    STATUS : BOOL; // Set to TRUE if the registry operation was successful, or FALS
END_VAR
VAR
    T_CAN_COBID_Buffer : T_CAN_COBID_Buffer; // Buffer to hold the COBID to be regi
END_VAR

T_CAN_COBID_Buffer.ID[T_CAN_COBID_Buffer.Length] := T_CAN_COBID.  ID;
T_CAN_COBID_Buffer.Length := T_CAN_COBID_Buffer.Length + 1;
STATUS := TRUE;

END_FUNCTION_BLOCK

FUNCTION_BLOCK CAN_Clear_Rcv_Message
VAR_INPUT
    T_CAN_COBID : T_CAN_COBID; // Struct holding the COBID to be deleted
END_VAR
VAR_OUTPUT
    STATUS : BOOL; // Set to TRUE if the deletion operation was successful, or FALS
END_VAR
VAR
    i : INT := 0; // Loop variable
BEGIN

STATUS := FALSE;

// Loop through the registered COBIDs to find the one to be deleted
FOR i := 0 TO T_CAN_COBID_Buffer.Length - 1 DO
    IF T_CAN_COBID_Buffer.ID[i] = T_CAN_COBID.cobid THEN
        // Found the COBID to be deleted; shift the remaining COBIDs in the buffer
        FOR j := i TO T_CAN_COBID_Buffer.Length - 2 DO
            T_CAN_COBID_Buffer.ID[j] := T_CAN_COBID_Buffer.ID[j+1];
        END_FOR
        T_CAN_COBID_Buffer.Length := T_CAN_COBID_Buffer.Length - 1;
        STATUS := TRUE;
        EXIT; // Exit the loop
    END_IF
END_FOR

END_FUNCTION_BLOCK
```

These functions assume that the T_CAN_COBID_Buffer structure is defined elsewhere in the program, and that it holds the list of registered COBIDs. The CAN_Set_Rcv_Message function simply adds the specified COBID to the buffer, while the CAN_Clear_Rcv_Message function loops through the buffer to find and remove the specified COBID. If the COBID is found and removed successfully, the STATUS output is set to TRUE; otherwise it is set to FALSE.