Hello,

I am working on a project involving CAN communication on STM32F429I-DISCO board. I have successfully ported CANopennode to that platform (using HAL drivers since there is STM32CubeMX support). I can see messages on PCAN being transmitted from a CAN1 (Master). Unfortunately I am not able to do the same with CAN2 (slave). It simply doesn't transmit any message at all. As a CAN transceiver I am using SN65HVD230 (STM32F4 Discovery Shield), code is written for Interrupt Transmit. While trying to transmit through CAN2 it goes into **HAL_CAN_ErrorCallback**. Log below:

```
**********************************************************
STM32F429ZI init!
CAN1_Init...
----------------- CAN1 initialized and ready for use ----------------------------
CAN2_Init...
----------------- CAN2 initialized and ready for use ----------------------------


Sending by CAN1
HAL_CAN_TxCpltCallback - Transmitted properly...


Receiving by CAN2
hcan->pRxMsg->StdId = 1
hcan->pRxMsg->Data[0] = 0
hcan->pRxMsg->Data[1] = 1
hcan->pRxMsg->Data[2] = 2
hcan->pRxMsg->Data[3] = 3
hcan->pRxMsg->Data[4] = 4
hcan->pRxMsg->Data[5] = 5
hcan->pRxMsg->Data[6] = 6
hcan->pRxMsg->Data[7] = 7


/* The part when I am trying to send with CAN2 */
HAL_CAN_ErrorCallback - Transmission failure...
Problem with CAN2...
HAL_CAN_GetError = 7 | hex = 7
HAL_CAN_ErrorCallback - Transmission failure...
Problem with CAN2...
HAL_CAN_GetError = 7 | hex = 7
HAL_CAN_ErrorCallback - Transmission failure...
Problem with CAN2...
HAL_CAN_GetError = 7 | hex = 7
HAL_CAN_ErrorCallback - Transmission failure...
Problem with CAN2...
HAL_CAN_GetError = 7 | hex = 7
HAL_CAN_ErrorCallback - Transmission failure...
...
```

Solutions I have tried:
- I have been switching CAN transceiver boards between CAN1 & CAN2,
- Change PIN configuration in STM32CubeMX for CAN2,
- I have compared live registers of CAN1 & CAN2 while debugging in Keil (nothing suspicious),
- I have checked Pinout states with Logic Analyzer, I see transmit on CAN1 but not on CAN2,
- Tested the code on another stm32f429 boards to exclude manufacturer error, result the same,
- I went through Reference Manual for that processor trying to find differences between CAN1 & CAN2, difference in setup, register values...

CAN2_RX is working, while sending data on CAN1 I see it on CAN2_RX. I tried LOOPBACK mode to see if there might be problem on CAN2 but it went fine.

The init structure of CAN1 & CAN2 is the same (prescaler, SJW, BS1, BS2 etc...).

---------- EDIT StdPeriph ----------
To exclude that problem occurs only on HAL drivers I have tested CAN transmit using StdPeripherial Drivers. I went through thread:
[DEAD LINK /public/STe2ecommunities/mcu/Lists/cortex_mx_stm32/Flat.aspx?RootFolder=/public/STe2ecommunities/mcu/Lists/cortex_mx_stm32/Can-Bus%20Error&FolderCTID=0x01200200770978C69A1141439FE559EB459D7580009C4E14902C3CDE46A77F0FFD06506F5B&currentviews=141]https://my.st.c
RootFolder=/public/STe2ecommunities/mcu/Lists/cortex_mx_stm32/Can-Bus%20Error&FolderCTID=0x01200200770978C69A1141439FE559EB459D7580009C4E14902C3CDE46A77F0FFD06506F5B&currentviews=141

Code is almost identical:

```
//  CAN1 PD0 (RX), PD1 (TX) J2, J3 ON
//  CAN2 PB12 (RX), PB13 (TX) board miswired RX/TX switched

#include <stm32f4xx.h>                  /* STM32F4xx Definitions             */

#include <stdio.h>
#include <stdlib.h>
#include <string.h> // memset

//*****************************************************************************

void RCC_Configuration(void)
{
  /* --------------------------- System Clocks Configuration -----------------*/
  /* Enable CAN clocks */
  RCC_APB1PeriphClockCmd(RCC_APB1Periph_CAN1 | RCC_APB1Periph_CAN2, ENABLE);

  /* GPIO[D] clock enable, B already enabled by USART3 */
  RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOD, ENABLE);
}

//*****************************************************************************
```

```
void GPIO_Configuration(void)
{
  GPIO_InitTypeDef GPIO_InitStructure;

  /*------------------------- GPIO Configuration ----------------------------*/
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF;
  GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
  GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;

  /* Configure CAN RX and TX pins */
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0 | GPIO_Pin_1;
  GPIO_Init(GPIOD, &GPIO_InitStructure);

  /* Connect CAN pins to AF */
  GPIO_PinAFConfig(GPIOD, GPIO_PinSource0, GPIO_AF_CAN1); // CAN1_RX
  GPIO_PinAFConfig(GPIOD, GPIO_PinSource1, GPIO_AF_CAN1); // CAN1_TX

  /* Configure CAN RX and TX pins */
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_12 | GPIO_Pin_13;
  GPIO_Init(GPIOB, &GPIO_InitStructure);

  /* Connect CAN pins to AF */
  GPIO_PinAFConfig(GPIOB, GPIO_PinSource12, GPIO_AF_CAN2); // CAN2_RX
  GPIO_PinAFConfig(GPIOB, GPIO_PinSource13, GPIO_AF_CAN2); // CAN2_TX
}

//***************************************************************************

void CAN_Configuration(void)
{
    int result=0;
  RCC_ClocksTypeDef     RCC_Clocks;
  CAN_InitTypeDef       CAN_InitStructure;
  CAN_FilterInitTypeDef CAN_FilterInitStructure;

  RCC_GetClocksFreq(&RCC_Clocks);

  /* CAN register init */
  CAN_DeInit(CAN1);
  CAN_DeInit(CAN2);

  CAN_StructInit(&CAN_InitStructure);

  /* CAN cell init */
  CAN_InitStructure.CAN_TTCM = DISABLE;
  CAN_InitStructure.CAN_ABOM = DISABLE;
  CAN_InitStructure.CAN_AWUM = DISABLE;
  CAN_InitStructure.CAN_NART = DISABLE;
  CAN_InitStructure.CAN_RFLM = DISABLE;
  CAN_InitStructure.CAN_TXFP = DISABLE;
  CAN_InitStructure.CAN_Mode = CAN_Mode_Normal;

  /* quanta 1+6+7 = 14, 14 * 24 = 336, 42000000 / 336 = 125000 */
  /* CAN Baudrate = 125Kbps (CAN clocked at 42 MHz) Prescale = 24 */

  /* Requires a clock with integer division into APB clock */

  CAN_InitStructure.CAN_SJW = CAN_SJW_1tq; // 1+6+7 = 14, 1+14+6 = 21, 1+15+5 = 21
  CAN_InitStructure.CAN_BS1 = CAN_BS1_16tq;
  CAN_InitStructure.CAN_BS2 = CAN_BS2_4tq;
  CAN_InitStructure.CAN_Prescaler = RCC_Clocks.PCLK1_Frequency / (21 * 125000); // quanta by baudrate
    printf("RCC_Clocks.PCLK1_Frequency = %d\n\r", RCC_Clocks.PCLK1_Frequency);

  result = CAN_Init(CAN1, &CAN_InitStructure);
    printf("CAN1 Init result = %d\n\r", result);
  result = 0;
    result = CAN_Init(CAN2, &CAN_InitStructure);
    printf("CAN2 Init result = %d\n\r", result);

  /* CAN filter init */
  CAN_FilterInitStructure.CAN_FilterMode = CAN_FilterMode_IdMask; // IdMask or IdList
  CAN_FilterInitStructure.CAN_FilterScale = CAN_FilterScale_32bit; // 16 or 32

  CAN_FilterInitStructure.CAN_FilterIdHigh      = 0x0000; // Everything, otherwise 11-bit in top bits
  CAN_FilterInitStructure.CAN_FilterIdLow       = 0x0000;
  CAN_FilterInitStructure.CAN_FilterMaskIdHigh  = 0x0000;
  CAN_FilterInitStructure.CAN_FilterMaskIdLow   = 0x0000;

  CAN_FilterInitStructure.CAN_FilterFIFOAssignment = CAN_FIFO0; // Rx
  CAN_FilterInitStructure.CAN_FilterActivation = ENABLE;

  CAN_FilterInitStructure.CAN_FilterNumber = 0; // CAN1 [ 0..13]

  CAN_FilterInit(&CAN_FilterInitStructure);

  CAN_FilterInitStructure.CAN_FilterNumber = 14; // CAN2 [14..27]

  CAN_FilterInit(&CAN_FilterInitStructure);
}

//***************************************************************************

void CAN2RX(void)
{
  CanRxMsg RxMessage;

  if (CAN_MessagePending(CAN2, CAN_FIFO0))
```

```
  {
    memset(&RxMessage, 0, sizeof(RxMessage));

    /* receive */
    CAN_Receive(CAN2, CAN_FIFO0, &RxMessage);

    printf("CAN2_RX %04X - %02X - %02X %02X %02X %02X %02X %02X %02X %02X\n",
      RxMessage.StdId, RxMessage.DLC,
      RxMessage.Data[0],RxMessage.Data[1],RxMessage.Data[2],RxMessage.Data[3],
      RxMessage.Data[4],RxMessage.Data[5],RxMessage.Data[6],RxMessage.Data[7]);
  }
}

//*****************************************************************************


void CAN1RX(void)
{
  CanRxMsg RxMessage;

  if (CAN_MessagePending(CAN1, CAN_FIFO0))
  {
    memset(&RxMessage, 0, sizeof(RxMessage));

    /* receive */
    CAN_Receive(CAN1, CAN_FIFO0, &RxMessage);

    printf("CAN1_RX %04X - %02X - %02X %02X %02X %02X %02X %02X %02X %02X\n",
      RxMessage.StdId, RxMessage.DLC,
      RxMessage.Data[0],RxMessage.Data[1],RxMessage.Data[2],RxMessage.Data[3],
      RxMessage.Data[4],RxMessage.Data[5],RxMessage.Data[6],RxMessage.Data[7]);
  }
}

//*****************************************************************************

void CAN1TX(void)
{
    int i=0;

  CanTxMsg TxMessage;
    uint8_t TransmitMailbox = 0;

  // transmit */
  TxMessage.StdId = 0x123;
  TxMessage.ExtId = 0x00;
  TxMessage.RTR = CAN_RTR_DATA;
  TxMessage.IDE = CAN_ID_STD;
  TxMessage.DLC = 8;


    for(i = 0; i < 8; i++) {
        TxMessage.Data[i] = i;
    }

    TransmitMailbox = CAN_Transmit(CAN1, &TxMessage);

    i = 0;
    while((CAN_TransmitStatus(CAN1, TransmitMailbox) != CANTXOK) && (i != 0xFFFFFF)) // Wait on Transmit
    {
        i++;
        CAN2RX(); // Pump RX
    }


    CAN2RX();
}

//*****************************************************************************

void CAN2TX(void)
{
    int i = 0;
  CanTxMsg TxMessage;
    uint8_t TransmitMailbox = 0;

  // transmit */
  TxMessage.StdId = 0x1234;
  TxMessage.ExtId = 0x00;
  TxMessage.RTR = CAN_RTR_DATA;
  TxMessage.IDE = CAN_ID_STD;
  TxMessage.DLC = 8;

    for(i = 0; i < 8; i++) {
        TxMessage.Data[i] = i;
    }


    i = 0;
    while((CAN_TransmitStatus(CAN1, TransmitMailbox) != CANTXOK) && (i != 0xFFFFFF)) // Wait on Transmit
    {
        i++;
        CAN1RX(); // Pump RX
    }


}
```

```
//*****************************************************************************
```

Init:

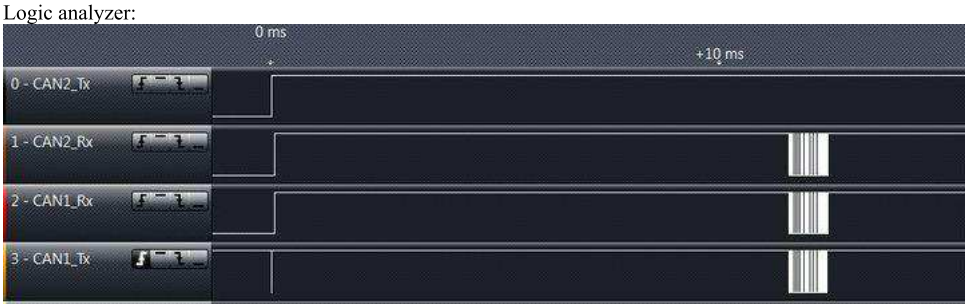
```
SystemInit();
NVIC_PriorityGroupConfig(NVIC_PriorityGroup_4);
init_USART3();                          /* USART3  Initialization             */
RCC_Configuration();
GPIO_Configuration();
CAN_Configuration();

printf("CAN1 send\n\r");
CAN1TX();
printf("CAN2 send\n\r");
CAN2TX();
```

Log from terminal:

```
RCC_Clocks.PCLK1_Frequency = 42000000
CAN1 Init result = 1
CAN2 Init result = 1
CAN1 send
CAN2_RX 0123 - 08 - 00 01 02 03 04 05 06 07
CAN2 send
```

Logic analyzer:



For the moment it doesn't work on StdPeriph & HAL. No solution for now.

---------- EDIT 08.12.2014 ----------
I have changed CAN2 pins to PB6 & PB5 (TX, RX) as it was originally. It works in LOOPBACK mode but not in NORMAL. I can't get acknowledgement on CAN2 and ACK error is indicated.

I can't find solution for the issue. I hope somebody has an idea.
Thank you for any help!