Bluetooth communicatie met app loopt fout waardoor FET’s kapot gaan.

Mogelijke oorzaken:

Overflow input buffer

Afhandeling inkomend bericht fout

Te nemen acties

Afhandeling inkomende berichten controleren en waar nodig verbeteren

Watchdog routine is nu niet aanwezig

Jeffrey,

het feit date er een overrun kan plaatsvinden is niet goed. hoe groot staat de FiFo van de uproc? Dan wordt de linbus\_run niet snel genoeg aan geroepen

Je zal eigenlijk een SCI interrupt handler moeten maken. Dat hoeft niet moeilijk te zijn, klein cyclisch buffer vullen en deze in de linbus\_run leeghalen.

Ik zie hier ook dat je over de grens van je buffer kan lopen:

            case Linbus\_State\_Read\_Length:  
                dLinbusVars.checksum ^= dLinbusVars.received;  
                dLinbusData.length = dLinbusVars.received;                                // Hier controleren of groter dan buffer! als geval state naar idle zetten  
                dLinbusVars.bufferIndex = 0;  
                dLinbusVars.state = Linbus\_State\_Read\_Data;  
            break;  
  
            case Linbus\_State\_Read\_Data:  
                dLinbusVars.checksum ^= dLinbusVars.received;  
                dLinbusData.data[dLinbusVars.bufferIndex] = dLinbusVars.received;  
                dLinbusVars.bufferIndex++;  
                if (dLinbusVars.bufferIndex >= dLinbusData.length)  
                    dLinbusVars.state = Linbus\_State\_Read\_Checksum;  
            break;

Bij de eerste state (LinBus\_State\_Read\_Length) kan je testen of de ontvangen lengte groter is dan buffer (dlinbusData)  als dat het geval is de state resetten naar (LinBus\_State\_Idle)

het resetten van de overrun flag is sowieso een goede zaak

ik zou van het leeghalen van de FiFo een while (…); maken

    if (dLinbusVars.fifoStatus > SCI\_FifoStatus\_Empty)

Maak daar van:

    while (dLinbusVars.fifoStatus > SCI\_FifoStatus\_Empty)

Dan lepel je de hele fifo leeg in een slag zonder weer de hele main lus door te moeten

Succes!

Hans

**Van:** Jeffrey Vos [mailto:[Jeffrey@idbike.com](mailto:Jeffrey@idbike.com)]   
**Verzonden:** dinsdag 28 januari 2020 10:59  
**Aan:** [hansk@inducon.eu](mailto:hansk@inducon.eu)  
**Onderwerp:** SCI Overflow

Hallo Hans,

Zou je even kunnen kijken of het onderstaande gaat werken?

De SCI heeft een overflow vlag (zie bijlage). De pdf is het hele document.

Als ik het goed begrijp kan ik deze bit toch ook gewoon checken buiten een interrupt?

Voordat ik het bericht ga binnenhalen check ik of deze overflow bit hoog is, zoja dan reset ik de sci. Hier onder te zien in de functie Linbus\_run:

Onder de Linbus\_run functie staat de getRxFifoStatus- en de clearRxFifoOvf functie.

void Linbus\_run(CTRL\_Handle ctrlHandle,HAL\_Handle halHandle)  
{  
    TxLinbushandler(halHandle);  
  
  
    if (SCI\_isError(halHandle->sciHandle))  
    {  
        SCI\_reset(halHandle->sciHandle);  
        SCI\_enable(halHandle->sciHandle);  
   }  
    dLinbusVars.fifoStatus = SCI\_getRxFifoStatus(halHandle->sciHandle);  
  
    // VosJe 27Jan2020 - Code below added to prevent buffer overflow  
    if(dLinbusVars.fifoStatus & 0x8000)  
    {  
        SCI\_clearRxFifoOvf(halHandle->sciHandle);  
        SCI\_reset(halHandle->sciHandle);  
        SCI\_enable(halHandle->sciHandle);  
    }  
    //////////////////  
  
    if (dLinbusVars.fifoStatus > SCI\_FifoStatus\_Empty)  
    {  
        dLinbusVars.received = (unsigned char) SCI\_getData(halHandle->sciHandle);  
        switch (dLinbusVars.state)  
        {  
            case Linbus\_State\_Idle:  
                if (dLinbusVars.received == 85) //0x55  
                {  
                    dLinbusVars.available = false;  
                    dLinbusVars.state = Linbus\_State\_Read\_Target;  
                    dLinbusVars.checksum = 0;  
                }  
            break;  
  
            case Linbus\_State\_Read\_Target:  
                dLinbusVars.checksum ^= dLinbusVars.received;  
                dLinbusVars.state = Linbus\_State\_Read\_Source;  
                dLinbusVars.ignore = (dLinbusVars.received != 0);  
            break;  
  
            case Linbus\_State\_Read\_Source:  
                dLinbusVars.checksum ^= dLinbusVars.received;  
                dLinbusVars.state = Linbus\_State\_Read\_Length;  
                if (!dLinbusVars.ignore)  
                    RemoteAddress = dLinbusVars.received;  
  
            break;  
  
            case Linbus\_State\_Read\_Length:  
                dLinbusVars.checksum ^= dLinbusVars.received;  
                dLinbusData.length = dLinbusVars.received;  
                dLinbusVars.bufferIndex = 0;  
                dLinbusVars.state = Linbus\_State\_Read\_Data;  
            break;  
  
            case Linbus\_State\_Read\_Data:  
                dLinbusVars.checksum ^= dLinbusVars.received;  
                dLinbusData.data[dLinbusVars.bufferIndex] = dLinbusVars.received;  
                dLinbusVars.bufferIndex++;  
                if (dLinbusVars.bufferIndex >= dLinbusData.length)  
                    dLinbusVars.state = Linbus\_State\_Read\_Checksum;  
            break;  
  
            case Linbus\_State\_Read\_Checksum:  
                if ((dLinbusVars.received == dLinbusVars.checksum)&&!dLinbusVars.ignore)  
                {  
                    // Message has been received completely.  
                    //if (dDisplayVars.SettingsState != DisplaySettingsState\_Idle)  
                        dProgramVars.timers[2] = 0;  // force fast update  
                   DecodeLinBusData(ctrlHandle, halHandle,dLinbusData.data,dLinbusData.length);  
                }  
                dLinbusVars.state = Linbus\_State\_Idle;  
            break;  
        }  
    }  
}

SCI\_FifoStatus\_e SCI\_getRxFifoStatus(SCI\_Handle sciHandle)  
{  
  SCI\_Obj \*sci = (SCI\_Obj \*)sciHandle;  
  SCI\_FifoStatus\_e status;  
  
  // get the status  
  status = (SCI\_FifoStatus\_e)(sci->SCIFFRX & SCI\_SCIFFRX\_FIFO\_ST\_BITS);  
  
  return(status);  
} // SCI\_getRxFifoStatus() function

void SCI\_clearRxFifoOvf(SCI\_Handle sciHandle)  
{  
  SCI\_Obj \*sci = (SCI\_Obj \*)sciHandle;  
  
  
  // set the bits  
  sci->SCIFFRX |= SCI\_SCIFFRX\_FIFO\_OVFCLR\_BITS;  
  
  return;  
} // end of SCI\_clearRxFifoOvf() function