We have a DP620 Display and a MC024-020 Controller. Introduced onto the canbus is a load cell digitiser(DSCSCOP by MANTRACOURT in the UK) sending data in a canopen protocol. This data is received into RX in the IEEE 754 format (single precision 32bit). We have no problems with communication, TX sends a start up message and node id and data streams in however it is incorrect.

Currently we have this system on two different PLC’s and functioning:

**Parker IQAN**



Significand Bits 22 - 0 =

(Bit0 / 8388608) + ((Bit1 / 4194304) + ((Bit2 / 2097152) + ((Bit3 / 1048576) + ((Bit4 / 524288) + ((Bit5 / 262144) + ((Bit6 / 131072) + ((Bit7 / 65536) + ((Bit8 / 32768) + ((Bit9 / 16384) + ((Bit10 / 8192) + ((Bit11 / 4096) + ((Bit12 / 2048) + ((Bit13 / 1024) + ((Bit14 / 512) + ((Bit15 / 256) + ((Bit16 / 128) + ((Bit17 / 64) + ((Bit18 / 32) + ((Bit19 / 16) + ((Bit20 / 8) + ((Bit21 / 4) + ((1 + 0) + (Bit22 / 2))))))))))))))))))))))).

Exponent Field Bits 30 – 23 =

((Bit30 \* 128) + ((Bit29 \* 64) + ((Bit28 \* 32) + ((Bit27 \* 16) + ((Bit26 \* 8) + ((Bit25 \* 4) + ((Bit23 \* 1) + (Bit24 \* 2)))))))) – 127

Sign Bit31 =

if Bit31 then -1 else 1

Emulsion Load Cell Output(SYS) =

(Sign Bit31 \* Significand Bits 22 - 0) \* PowerOf(2, Exponent Field Bits 30 - 23)

**Codesys**

VAR

 FIRST\_SCAN:BOOL :=TRUE;

 MY\_CAN\_TX\_ENH\_CYCLIC:CAN\_TX\_ENH\_CYCLIC;

 MY\_CAN\_ENABLE:CAN\_ENABLE;

 MY\_CAN\_RX:CAN\_RX;

 CAN\_ENABLE\_RESULT:BYTE ;

 RESULT\_STARTUP\_MESSAGE: BYTE;

 LOADCELL\_OUTPUT: ARRAY [0..7] OF BYTE;

 LOADCELL\_OUTPUT\_TEMP : ARRAY [0..3] OF BYTE;

 RESULT\_LOAD\_CELL\_OUTPUT: BYTE;

 STARTUP\_TIME: TIME := T#1000ms;

 LOAD\_CELL\_POINTER\_REAL :POINTER TO REAL;

 LOAD\_CELL\_OUTPUT\_mV\_PER\_V :REAL;

 LOAD\_CELL\_OUTPUT\_MSG\_COUNTER : UINT :=0;

 STARTUP\_CAN\_DATA: ARRAY [0..7] OF BYTE; (\* STARTUP DATA BYTE 0 =1 COMMAND START NOTE, BYTE 2 IS THE NODE ID \*)

 FILTER\_ARRAY\_SIZE : INT := 25;

 LOAD\_CELL\_OUTPUT\_FILTER\_ARRAY: ARRAY[1..25] OF REAL;

 FILTER\_ARRAY\_INDEX : INT :=1;

 LOAD\_CELL\_OUTPUT\_FILTERED: REAL;

 I : INT ;

 LOAD\_CELL\_SUM:REAL;

END\_VAR

(\* PAVELINE LOAD CELL DIGITIZER FOR EMULSION VESSEL LOAD CELLS. (NODE ID= 120) \*)

(\* ON START UP... \*)

IF FIRST\_SCAN THEN

 LOAD\_CELL\_POINTER\_REAL := ADR(LOADCELL\_OUTPUT\_TEMP);

 STARTUP\_CAN\_DATA[0] := 1 ; (\* CAN MESSAGE TO START UP DIGITIZER \*)

 STARTUP\_CAN\_DATA[1] := NODE\_ID;

 STARTUP\_CAN\_DATA[2] :=0;

 STARTUP\_CAN\_DATA[3] :=0;

 STARTUP\_CAN\_DATA[4] :=0;

 STARTUP\_CAN\_DATA[5] :=0;

 STARTUP\_CAN\_DATA[6] :=0;

 STARTUP\_CAN\_DATA[7] :=0;

END\_IF

FIRST\_SCAN := FALSE;

(\* SEND CAN MESSAGE TO DIGITIZER TO START UP. SEND EVERY SECOND JUST IN CASE IT IS POWERED THEN RESTARTED \*)

MY\_CAN\_TX\_ENH\_CYCLIC(

 ENABLE:= TRUE,

 CHANNEL:= CAN\_CHANNEL,

 ID:= 0,

 Extended:=FALSE ,

 DataLengthCode:=2 ,

 DATA:=STARTUP\_CAN\_DATA ,

 PERIOD:= T#1000ms ,

 RESULT=> );

(\* RECIEVE CAN MESSAGE FROM DIGIZER. VALUE IS THE mV/V OUTPUT FROM THE LOAD CELL. LOAD CELL CONTINUSOULY TRANSMITS THIS VALUE \*)

(\* CAN ID = 504 =384 (180hex)+NodeID \*)

MY\_CAN\_RX(

 ENABLE:= TRUE ,

 CHANNEL:=CAN\_CHANNEL ,

 ID:=384+NODE\_ID ,

 DATA=>LOADCELL\_OUTPUT ,

 RESULT=>RESULT\_LOAD\_CELL\_OUTPUT);

(\* WHEN A CAN MESSAGE FOR LOAD CELLS IS RECEIVED GRAB THE LOAD CELL OUTPUT \*)

IF RESULT\_LOAD\_CELL\_OUTPUT = 1 THEN

 (\* LOAD CELL OUTPUT IS STORED AS A 4 BYTE REAL NUMBER. ORDER IS REVERSED... \*)

 LOAD\_CELL\_OUTPUT\_MSG\_COUNTER := LOAD\_CELL\_OUTPUT\_MSG\_COUNTER +1;

 LOADCELL\_OUTPUT\_TEMP[0] := LOADCELL\_OUTPUT[3];

 LOADCELL\_OUTPUT\_TEMP[1] := LOADCELL\_OUTPUT[2];

 LOADCELL\_OUTPUT\_TEMP[2] := LOADCELL\_OUTPUT[1];

 LOADCELL\_OUTPUT\_TEMP[3] := LOADCELL\_OUTPUT[0];

 LOAD\_CELL\_OUTPUT\_mV\_PER\_V :=LOAD\_CELL\_POINTER\_REAL^;

 LOAD\_CELL\_OUTPUT :=LOAD\_CELL\_OUTPUT\_mV\_PER\_V ;

 (\* AVERAGE THE LOAD CELL VALUE OVER 25 READINGS \*)

 LOAD\_CELL\_OUTPUT\_FILTER\_ARRAY[FILTER\_ARRAY\_INDEX] := LOAD\_CELL\_OUTPUT;

 FILTER\_ARRAY\_INDEX := FILTER\_ARRAY\_INDEX + 1;

 IF (FILTER\_ARRAY\_INDEX >FILTER\_ARRAY\_SIZE ) THEN

 FILTER\_ARRAY\_INDEX := 1;

 END\_IF

 LOAD\_CELL\_SUM :=0.0;

 FOR I:=1 TO FILTER\_ARRAY\_SIZE DO

 LOAD\_CELL\_SUM := LOAD\_CELL\_OUTPUT\_FILTER\_ARRAY[I] + LOAD\_CELL\_SUM;

 END\_FOR;

 LOAD\_CELL\_OUTPUT\_FILTERED:=LOAD\_CELL\_SUM / INT\_TO\_REAL(FILTER\_ARRAY\_SIZE);

END\_IF

As you can see Parker IQAN unpacks the data into individual bits (32) then the IEEEC 754 equation assembles it back to a decimal number.

With Co De Sys (IFM Electronic) a pointer is used as seen above to get 32 bits to there original decimal number.

Then in both cases it is multiplied by a scale factor to give us weight in kilograms.

We have faith in the IEEEC 754 equation you see below(it is a direct copy of the Parker IQAN one) and we feel the problem may lie in the way we have attempted to isolate the 32 bits using SPLITS.





The above module is quite verbose and a little untidy. We understand there may be better ways to do it. We are quite new to the PLUS+1 environment.

The second problem we envisage is that our decimal number or float will not be accepted as PLUS+1 only does integers.

Below is another attempt, the function block we made may be right or not but we still feel the use of combine to put the 32 bit canbus message back together may not be correct.



Any assistance or guidance is very much appreciated