LFR-FSW - Bug #448

R3 *** BP1 : valeurs VPHI et SX aberrantes

26/06/2015 10:31 AM - bruno katra

Status:	Closed	**Start date:**	26/06/2015
Priority:	Urgent	**Due date:**	
Assignee:	bruno katra	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour
revision:	r79		

Description

Les valeurs ne sont pas pertinentes.
Piste prioritaire :
- ~~Pb au niveau DECOM lié au code de Thomas : la variable nbitsig (basic_parameter.h) est toujours égale à 10 bits alors que VPHI et SX ont des parties flottantes SIGNIFICAND sur 8 bits. Normal ?~~

CONTEXTE :

- DECOM r76
- EM1
- FSW 3.0.0.7
- Analog discovery en conf "6on2"
- Spw USB

History

#1 - 26/06/2015 10:44 AM - bruno katra

- Status changed from New to Feedback

- % Done changed from 0 to 30

- revision changed from r0 to r76

Pb trouvé dans la DECOM : reliquat d'un calcul des BP1 R2 dans une des méthodes .
Corrigé en r77 et relivré.

#2 - 26/06/2015 11:47 AM - thomas chust

- Assignee changed from thomas chust to bruno katra

Dans le fichier .h la variable nbitsig n'ai utilisé que pour les puissances spectrales PE et PB. Pour SX et VPHI le SIGNIFICAND est codé sur 8 bits comme indiqué dans l'ICD R3.
Mais effectivement mes commentaires (copiés collés ...) sont ambigüe ...

```
significand = frexpf(e_cross_b_re, &exponent); // 0.5 <= significand < 1
                                                // ReaSX = significand * 2^exponent
        if (exponent < expmin) { // value should be >= 0.5 * 2^expmin
          exponent = expmin;
          significand = 0.5;     // min value that can be recorded
        }
        if (exponent > expmax) { // value should be <  0.5 * 2^(expmax+1)
          exponent = expmax;
          significand = 1.0;     // max value that can be recorded
        }
        if (significand  0) {  // in that case exponent  0 too
          exponent = expmin;
          significand = 0.5;     // min value that can be recorded
        }
```

```
lfr_bp1[i*NB_BYTES_BP1+8] = (uint8_t) ((significand*2-1)*255 + 0.5); // Shift and cast into a 8-bit uint8_t wi
th rounding
                                                                     // where all bits are used (0, ..
., 255)
        tmp_uint8 = (uint8_t) (exponent-expmin); // Shift and cast into a 8-bit uint8_t where
```

```
                                                    // just the first nbitexp bits are used (0, ..., 2^nbitexp-1)
#ifdef DEBUG_TCH
        printf("|ReaSX|     : %16.8e\n",e_cross_b_re);
        printf("significand : %16.8e\n",significand);
        printf("exponent    : %d\n"    ,exponent);
        printf("tmp_uint8         for ReaSX exponent    : %d\n",tmp_uint8);
#endif
        lfr_bp1[i*NB_BYTES_BP1+7] = lfr_bp1[i*NB_BYTES_BP1+7] | tmp_uint8; // Record these nbitexp bits in the
 nbitexp first bits
                                                                           // (from the right to the left) of
lfr_bp1[i*NB_BYTES_BP1+7]
```

#3 - 26/06/2015 12:42 PM - bruno katra

Ok alors je pense que je ne suis pas conforme dans la façon dont je decommute.

thomas chust wrote:

> Dans le fichier .h la variable nbitsig n'ai utilisé que pour les puissances spectrales PE et PB. Pour SX et VPHI le SIGNIFICAND est codé sur 8 bits comme indiqué dans l'ICD R3.
> Mais effectivement mes commentaires (copiés collés ...) sont ambigüe ...
>
> ```
> significand = frexpf(e_cross_b_re, &exponent); // 0.5 <= significand < 1
> // ReaSX = significand * 2^exponent
> if (exponent < expmin) { // value should be >= 0.5 * 2^expmin
> exponent = expmin;
> significand = 0.5;     // min value that can be recorded
> }
> if (exponent > expmax) { // value should be <  0.5 * 2^(expmax+1)
> exponent = expmax;
> significand = 1.0;     // max value that can be recorded
> }
> if (significand  0) {  // in that case exponent  0 too
> exponent = expmin;
> significand = 0.5;     // min value that can be recorded
> }
>
> lfr_bp1[i*NB_BYTES_BP1+8] = (uint8_t) ((significand*2-1)*255 + 0.5); // Shift and cast into a 8-bit uint8_t with rounding
> // where all bits are used (0, ..., 255)
> tmp_uint8 = (uint8_t) (exponent-expmin); // Shift and cast into a 8-bit uint8_t where
> // just the first nbitexp bits are used (0, ..., 2^nbitexp-1)
> #ifdef DEBUG_TCH
> printf("|ReaSX|     : %16.8e\n",e_cross_b_re);
> printf("significand : %16.8e\n",significand);
> printf("exponent    : %d\n"    ,exponent);
> printf("tmp_uint8         for ReaSX exponent    : %d\n",tmp_uint8);
> #endif
> lfr_bp1[i*NB_BYTES_BP1+7] = lfr_bp1[i*NB_BYTES_BP1+7] | tmp_uint8; // Record these nbitexp bits in the nbitexp first bits
> // (from the right to the left) of lfr_bp1[i*NB_BYTES_BP1+7]
> ```

#4 - 26/06/2015 02:29 PM - bruno katra

- % Done changed from 30 to 80

r78 : correction rangesig pour VPHI et SX.
Relivré.

#5 - 26/06/2015 07:39 PM - thomas chust

la r78 ne semble pas améliorer les choses. Et je ne trouve pas de bugg de mon coté (mon fichier .c qui calcul SX et VPHI, ou la façon dont je fais la vérification en idl) ? A suivre ...

#6 - 29/06/2015 12:15 PM - bruno katra

- Project changed from LFR-FSW to DECOM LFR

#7 - 29/06/2015 12:19 PM - bruno katra

*- Subject changed from R3 *** BP1 : valeurs VPHI et SX aberrantes to R2/R3 *** BP1 : valeurs VPHI et SX aberrantes*

r79 : pb de precision corrigé pour la R3 pour plusieurs valeurs BP1 et BP2. Doivent être calculés en double car la partie exp peut le necessiter dans certains cas. Sinon, des pbs de precision apparaissent : valeurs aberrantes.

!!! Le pb doit aussi être corrigé en R2 !!!

#8 - 29/06/2015 02:13 PM - bruno katra

- Status changed from Feedback to In Progress

- Assignee changed from bruno katra to thomas chust

Il y avait bien un pb de precision qui a été corrigé mais les valeurs aberrantes persistent.
Relecture complète du code de DECOM et comparaison avec calcul manuel sur des cas précis. A priori, il n'y a plus d'erreur dans la DECOM, les valeurs sont bien celles qui sont lues dans le datadump binaire ICD 3.9.

#9 - 29/06/2015 05:40 PM - bruno katra

- Project changed from DECOM LFR to LFR-FSW

#10 - 29/06/2015 06:04 PM - thomas chust

- File 2015_06_26_13_32_59_packet_record_NORMAL.1f2 added

- Assignee changed from thomas chust to paul leroy

- Priority changed from Normal to Urgent

- % Done changed from 80 to 50

- revision changed from r76 to r79

De mon coté il ne semble pas y avoir d'erreur non plus. Les valeurs pour SX et VPHI que LFR nous donne ne correspondent pas à ce que le programme basic_parameters calcule. Ces valeurs sont même parfois complètement aberrantes car très supérieures à 2^30 (voir le fichier joint pour un exemple).

Une piste: une des différence avec la R2 est qu'avec la R3 le tableau LFR_BP1_f0 a 2 octets de plus (passage de 9 à 11 octets). Es-tu bien sur que tu lis ce tableau avec les bonnes dimensions mises à jours? Les données SX et VPHI sont justement stockées dans les 4 derniers octets ...

#11 - 30/06/2015 11:15 AM - bruno katra

- Description updated

#12 - 30/06/2015 11:33 AM - bruno katra

*- Subject changed from R2/R3 *** BP1 : valeurs VPHI et SX aberrantes to R3 *** BP1 : valeurs VPHI et SX aberrantes*

#13 - 30/06/2015 11:59 AM - bruno katra

Salut Paul,
Je te réponds point par point:

Le 30/06/2015 11:26, Paul LEROY a écrit :

> Questions:
> => ça marche de façon nominale pour la R2? L'intégralité du paquet décommuté est correct pour tous les paramètres?

Oui jusqu'à présent les BP1 en R2 étaient correctes

> => en R3, les seuls paramètres que vous ne voyez pas correctement sont SX et VPHI? Tous les autres sont corrects, pour toutes les bins? Les BP2 sont corrects?

Oui seuls les SX et les VPHI sont incorrects. Les autres données BP1 sont correctes ainsi que tous les BP2.

> => Thomas, as-tu bien pris en compte le fait que la représentation MSB/LSB n'est pas la même sur LFR et sur ton PC (pour un float, les 4 octets sont inversés)

Oui. Et a priori cela n'intervient pas vraiment dans l'écriture en mémoire de SX et VPHI

?
Thomas

#14 - 30/06/2015 01:34 PM - paul leroy

Question: avez-vous fait tourner le calcul de Thomas sur les données matrices spectrales réelles, lorsque celles-ci sont livrées avec les BP1.

Si oui, il serait intéressant de comparer les valeur en hexa lues dans les BP1 reçu et les valeurs en hexa attendues (celle du calcul de référence de

Thomas affectuées sur les matrices spectrales reçues).

#15 - 30/06/2015 02:39 PM - thomas chust

Oui je lis les données ASM associées (toutes les 4 secondes aussi) et calcule sous idl ce que le programme C est sensé faire.
Et c'est bien là le problème. On observe pas ce qu'il faudrait ...

#16 - 30/06/2015 02:47 PM - paul leroy

Ca serait possible de faire le calcul avec le programme en c?
Ceci pour vérifier deux choses: ça marche avec des float (en supposant que linux ne fasse pas le calcul en douce en double), et aussi pour regarder les valeurs en hexa qu'on est censé voir dans les BP, et comparer avec ce qui est reçu.

#17 - 30/06/2015 03:36 PM - thomas chust

- File setup_sm_test2_R3.txt added

- File tab_test2_R3_v2.0.pdf added

- File sm_test2_R3.txt added

En fait c'est ce que j'avais déjà fait pour valider ma release R3 des BP1 et BP2. Ci-joint 3 fichiers en font la demonstration:
- sm_test2_R3.txt => la matrice spectrale que je vais lire et charger en C pour tester le bon calcul des BP
- setup_sm_test2_R3.txt => la façon dont cette matrice a été générée sous idl
- tab_test2_R3_v2.0.pdf => le résultat du test, c'est à dire tous les printf effectués par le programme C lorsqu'il calcule les BP1 et BP2
L'analyse de cela montre un accord parfait avec ce qui est attendu. En particulier pour le problème que nous avons, a minima:
- lfr_bp1[i*NB_BYTES_BP1+7] est l'endroit où se passe un problème : les 6 premiers bits qui code l'exposant de SX est actuellement (à bord de LFR) n'importe quoi.
- idem avec lfr_bp1[i*NB_BYTES_BP1+9] pour l'exposantr de VPHI

#18 - 30/06/2015 03:56 PM - paul leroy

OK. En hexa, serais-tu capable de dire si les deux bytes sont faux ou si un des deux est correct?

#19 - 30/06/2015 04:18 PM - thomas chust

A minima lfr_bp1[i*NB_BYTES_BP1+7] et lfr_bp1[i*NB_BYTES_BP1+9] sont mal retranscrits.
Concernant les octets stockant les valeurs significand lfr_bp1[i*NB_BYTES_BP1+8] lfr_bp1[i*NB_BYTES_BP1+10], valeurs comprises entre 0.5 et 1 je ne sais pas si elles sont fausses. Le contraire me surprendait mais ...

#20 - 01/07/2015 11:31 AM - paul leroy

- File sm_test2_R3_done_on_lfr.txt added

J'ai lancé le calcul directement sur LFR en utilisant les valeurs de la matrice que vous avez utilisé pour les tests. On trouve des différences dans les résultats pour les octets 7 et 9 du jeu de 11 bytes, j'ai l'impression que les valeurs 8 et 10 sont conformes.
Thomas, il fauudrait que tu passes chez moi pour regarder directement. Je joins la capture enregistrée pendant le test.

#21 - 01/07/2015 11:47 AM - bruno katra

paul leroy wrote:

> J'ai lancé le calcul directement sur LFR en utilisant les valeurs de la matrice que vous avez utilisé pour les tests. On trouve des différences dans les résultats pour les octets 7 et 9 du jeu de 11 bytes, j'ai l'impression que les valeurs 8 et 10 sont conformes.
> Thomas, il fauudrait que tu passes chez moi pour regarder directement. Je joins la capture enregistrée pendant le test.

Commentaire Bruno : cela semble cohérent avec les erreurs observées car selon Thomas, ces erreurs sont relatives à l'exposant du flottant qui est justement dans les octets 7 et 9.

#22 - 01/07/2015 02:27 PM - thomas chust

Merci Paul c'est un test concluant! Je crois avoir compris.
A priori c'est de ma faute: je n'ai pas pris garde de préaffecter une valeur nulle aux octects 7 et 9 alors que je les modifie via des opération logiques (et ou), ce qui n'était pas le cas pour le programme R2. Cela explique l'aspect aléatoire des résultats.

Je fais bientôt une modif et un push ...

#23 - 01/07/2015 03:15 PM - thomas chust

- Status changed from In Progress to Resolved

correction faite et pushée (version 2.3):
https://chust@hephaistos.lpp.polytechnique.fr/rhodecode/HG_REPOSITORIES/LPP/INSTRUMENTATION/USERS/CHUST/LFR_basic-parameters

#24 - 02/07/2015 08:52 AM - paul leroy

- File sm_test2_R3_done_on_lfr_fsw_3-0-0-8.txt added

- Assignee changed from paul leroy to bruno katra

J'ai relancé le test sur LFR et c'est concluant.
J'ai mis sur pc-instru la version 3.0.0.8 du logiciel de vol qui intègre la modification sur BP1_set.

#25 - 08/07/2015 10:40 AM - bruno katra

- Status changed from Resolved to Closed

- % Done changed from 50 to 100

DECOM r79 testée sur CTC-015 avec FSW 3.0.0.8 (SBM2 avec signaux + phase 90°) : Thomas a dépouillé et les valeurs sont OK.

Files

2015_06_26_13_32_59_packet_record_NORMAL.1f2	40.3 KB	29/06/2015	thomas chust
setup_sm_test2_R3.txt	10 KB	30/06/2015	thomas chust
tab_test2_R3_v2.0.pdf	21.2 KB	30/06/2015	thomas chust
sm_test2_R3.txt	805 Bytes	30/06/2015	thomas chust
sm_test2_R3_done_on_lfr.txt	2.13 KB	01/07/2015	paul leroy
sm_test2_R3_done_on_lfr_fsw_3-0-0-8.txt	6.22 KB	02/07/2015	paul leroy