DDR SFW Firmware

Presented to Christophe Deutsch,

Telops inc.

|  |  |
| --- | --- |
| Document name: | DDR SFW Firmware |
| Document number: | TEL-IRCDEV-00XXX |
| Version: | a |
| Date: | 12 March 2012 |

|  |
| --- |
| Proprietary Information  The information contained in this document is considered proprietary to Telops Inc. and should not be disclosed, copied, reproduced or distributed without the written permission of Telops Inc. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Prepared, reviewed, or approved by** | | **Function** | **Signature** | **Date** |
| P | Julien Roy | Developer |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Document Change Record

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Version | Reason for change | By | Approbation |
| 12 March 2012 | a | Première version | JRO |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

INSTRUCTIONS:

1. Les signets-documents sont situés sur la page frontispice, dans la zone illustrée sur la figure à gauche.
2. Pour faciliter l’édition des signets, sélectionner « signets » dans la zone « afficher » de l’onglet « affichage » de l’item « option » dans le menu « outils ». Ceci permet de voir les brackets comme dans la figure à gauche. S’assurer de taper le texte à l’intérieur des brackets.
3. Le « document name » en deux lignes aide la disposition de ce champ dans l’entête des pages.
4. La ligne « Type of Document » se remplace directement par, par exemple, « Cost Proposal »
5. La ligne « presented to … » ainsi que la suivante se remplacent directement.

Signets

Documents

1. Il est possible que le formattage des champs updatés automatiquement dans l’entête du document soit défait par une nouvelle entrée de signets-document.
2. Mettre la date manuellement (pas de champ automatique) dans le Document Change Record. Même commentaire sur l’Issue et révision.
3. SI VOUS AVEZ DES AMéLIORATIONS À CE TEMPLATE, SVP SUGGéREZ-LES MOI PLUTOT QUE DE PARTIR UN TEMPLATE ILLéGAL.

Table of Content

[Document Change Record ii](#_Toc322950952)

[Table of Content iii](#_Toc322950953)

[1 Introduction 1](#_Toc322950954)

[1.1 Définition du document 1](#_Toc322950955)

[1.2 Documents de référence 1](#_Toc322950956)

[1.3 Acronymes 1](#_Toc322950957)

[2 Implantation 2](#_Toc322950958)

[2.1 Paramètres des headers 2](#_Toc322950959)

[2.1.1 SFWPosition 2](#_Toc322950960)

[2.1.2 SFWEncoderAtExposureStart 2](#_Toc322950961)

[2.1.3 SFWEncoderAtExposureEnd 2](#_Toc322950962)

[2.2 SFWPosition 2](#_Toc322950963)

[2.3 SFWPositionSetpoint 3](#_Toc322950964)

[2.4 SFWSpeedSetpoint 3](#_Toc322950965)

[2.5 ExposureTime 4](#_Toc322950966)

[2.5.1 SFWMode 4](#_Toc322950967)

[2.5.2 EHDRIMode 4](#_Toc322950968)

[2.6 Trigger 4](#_Toc322950969)

[2.7 AEC 4](#_Toc322950970)

[2.8 Vidéo 5](#_Toc322950971)

[2.8.1 NTSC 5](#_Toc322950972)

[2.8.2 AGC 5](#_Toc322950973)

[2.9 Expansion board 5](#_Toc322950974)

[2.10 EEPROM 5](#_Toc322950975)

[2.11 SFW Feedback 6](#_Toc322950976)

[2.12 SFW Processing 6](#_Toc322950977)

[2.13 Contrôleur du moteur 7](#_Toc322950978)

[3 Implantation à faire 8](#_Toc322950979)

[3.1 Paramètres du header 8](#_Toc322950980)

[3.2 EEPROM 8](#_Toc322950981)

[3.3 AGC 8](#_Toc322950982)

[3.4 Calibration 8](#_Toc322950983)

[3.5 EHDRI 8](#_Toc322950984)

[3.6 AEC+ 8](#_Toc322950985)

[3.7 État de la caméra 8](#_Toc322950986)

[3.8 gEEPROMSFWPresent 9](#_Toc322950987)

[3.9 Autres détecteurs (plus tard) 9](#_Toc322950988)

[4 TESTS 10](#_Toc322950989)

[4.1 Faits 10](#_Toc322950990)

[4.1.1 Contrôle de la roue à filtre 10](#_Toc322950991)

[4.1.2 Synchronisation des encodeurs 10](#_Toc322950992)

[4.1.3 Écriture des headers 10](#_Toc322950993)

[4.1.4 Génération du trigger en synchronously rotating 11](#_Toc322950994)

[4.1.5 EHDRI et SFW 11](#_Toc322950995)

[4.2 À faire 11](#_Toc322950996)

[4.2.1 Génération du signal d’intégration (Hercules) 11](#_Toc322950997)

[4.2.2 Synchronisation du filtre 11](#_Toc322950998)

[4.2.3 Mises à jour temps réel 11](#_Toc322950999)

[4.2.4 AEC 11](#_Toc322951000)

[4.2.5 Centre des filtres 11](#_Toc322951001)

[4.2.6 Répétabilité du positionnement 11](#_Toc322951002)

[4.2.7 Stabilité de la SFW 12](#_Toc322951003)

[4.2.8 Global 12](#_Toc322951004)

[5 Modification du firmware 13](#_Toc322951005)

[5.1 SFW 13](#_Toc322951006)

[5.1.1 Source 13](#_Toc322951007)

[5.1.1.1 fw\_decoder.vhd 13](#_Toc322951008)

[5.1.1.2 SFW\_WatchDog.vhd 13](#_Toc322951009)

[5.1.1.3 SFW\_Ctrl.vhd 13](#_Toc322951010)

[5.1.1.4 SFW.bde 13](#_Toc322951011)

[5.1.2 Driver 14](#_Toc322951012)

[5.1.2.1 FW\_timer.h/.cpp 14](#_Toc322951013)

[5.1.2.2 maxonProcessing.h/.cpp 14](#_Toc322951014)

[5.1.2.3 RS232\_Maxon.h/.cpp 14](#_Toc322951015)

[5.1.2.4 EPOS\_Control.h/.cpp 14](#_Toc322951016)

[5.1.2.5 maxonUtilities.h/.cpp 14](#_Toc322951017)

[5.1.2.6 FWController.h/.cpp 14](#_Toc322951018)

[5.1.2.7 SFW\_Processing.h/.cpp 15](#_Toc322951019)

[5.1.2.8 SFW\_MathematicalModel.h/.cpp 15](#_Toc322951020)

[5.1.3 Testbench 16](#_Toc322951021)

[5.1.3.1 quadrature\_encoder.vhd 16](#_Toc322951022)

[5.2 TMI\_LL\_Histogram 16](#_Toc322951023)

[5.2.1 Source 16](#_Toc322951024)

[5.2.1.1 Histogram\_mcw 16](#_Toc322951025)

[5.2.1.2 ll\_tmi\_histogram\_1024.bde 16](#_Toc322951026)

[5.3 AEC 16](#_Toc322951027)

[5.3.1 Source 16](#_Toc322951028)

[5.3.1.1 AEC\_Ctrl.vhd 16](#_Toc322951029)

[5.3.1.2 AEC.bde 16](#_Toc322951030)

[5.3.2 Driver 16](#_Toc322951031)

[5.3.2.1 AEC.h/.cpp 16](#_Toc322951032)

[5.4 Image Sequencer 17](#_Toc322951033)

[5.4.1 Source 17](#_Toc322951034)

[5.4.1.1 IRC\_define.vhd (Common\_HDL) 17](#_Toc322951035)

[5.4.1.2 img\_sequencer\_Ctrl.vhd 17](#_Toc322951036)

[5.4.1.3 img\_exp\_time\_sm.vhd 17](#_Toc322951037)

[5.4.1.4 Image\_Sequencer.bde 17](#_Toc322951038)

[5.4.2 Driver 17](#_Toc322951039)

[5.4.2.1 Image\_Sequencer.h/.cpp 17](#_Toc322951040)

[5.5 FPA (isc0207) 18](#_Toc322951041)

[5.5.1 Source 18](#_Toc322951042)

[5.5.1.1 fpa\_intf\_cfg\_receiver.vhd 18](#_Toc322951043)

[5.5.1.2 fpa\_isc0207\_interface.bde 18](#_Toc322951044)

[5.5.2 Driver 18](#_Toc322951045)

[5.5.2.1 fpa\_intf.h/.cpp 18](#_Toc322951046)

[5.6 FPA (Hercules) 18](#_Toc322951047)

[5.6.1 Source 18](#_Toc322951048)

[5.6.1.1 Hercules\_COM\_Register.vhd 18](#_Toc322951049)

[5.6.1.2 Hercules\_HDRI\_Ram.vhd 18](#_Toc322951050)

[5.6.1.3 Hercules\_Header\_Extractor.vhd 19](#_Toc322951051)

[5.6.1.4 Hercules\_trig\_proc.vhd 19](#_Toc322951052)

[5.6.1.5 Trigger\_Filter.vhd 19](#_Toc322951053)

[5.6.1.6 Hercules\_exp\_time.vhd 19](#_Toc322951054)

[5.6.1.7 Hercules\_COM\_Controller.bde 19](#_Toc322951055)

[5.6.1.8 Hercules\_COM\_Manager.bde 19](#_Toc322951056)

[5.6.1.9 Hercules\_interface.bde 19](#_Toc322951057)

[5.6.2 Driver 20](#_Toc322951058)

[5.6.2.1 fpa\_intf.h/.cpp 20](#_Toc322951059)

[5.6.2.2 Hercules\_Processing.h/.cpp 20](#_Toc322951060)

[5.7 FRAME\_BUFFEr\_COADD 20](#_Toc322951061)

[5.7.1 Source 20](#_Toc322951062)

[5.7.1.1 WB\_CoAdd\_CTRL.vhd 20](#_Toc322951063)

[5.7.1.2 MAIN\_CTRL.vhd 20](#_Toc322951064)

[5.7.1.3 FBC\_CTRL.bde 21](#_Toc322951065)

[5.7.1.4 Frame\_Buffer\_CoAdd.bde 21](#_Toc322951066)

[5.7.1.5 CoAdd\_FullSyst\_Top.bde 21](#_Toc322951067)

[5.7.2 Driver 21](#_Toc322951068)

[5.7.2.1 Frame\_Buffer\_CoAdd.h/.cpp 21](#_Toc322951069)

[5.8 EHDRI 21](#_Toc322951070)

[5.8.1 Driver 21](#_Toc322951071)

[5.8.1.1 EHDRI\_Acquisition.h/.cpp 21](#_Toc322951072)

[5.9 Trig Ctrl 22](#_Toc322951073)

[5.9.1 Source 22](#_Toc322951074)

[5.9.1.1 Trigger\_ctler\_kernel.vhd 22](#_Toc322951075)

[5.9.1.2 fpa\_trigger\_define.vhd 22](#_Toc322951076)

[5.9.1.3 fpa\_trigger.bde 22](#_Toc322951077)

[5.9.1.4 fpa\_trigger\_ctler.bde 22](#_Toc322951078)

[5.9.2 Driver 22](#_Toc322951079)

[5.9.2.1 Trigger.h/.cpp 22](#_Toc322951080)

[5.10 Video 22](#_Toc322951081)

[5.10.1 Source 22](#_Toc322951082)

[5.10.1.1 AGC\_Ctrl.vhd 22](#_Toc322951083)

[5.10.1.2 AGC.bde 22](#_Toc322951084)

[5.10.1.3 Video\_top.bde 22](#_Toc322951085)

[5.11 HDR Merge 23](#_Toc322951086)

[5.11.1 Driver 23](#_Toc322951087)

[5.11.1.1 HDR\_Merge.h/.cpp 23](#_Toc322951088)

[5.12 AGC 23](#_Toc322951089)

[5.13 Top Level 23](#_Toc322951090)

[5.13.1 VHDL 23](#_Toc322951091)

[5.13.1.1 ROIC\_Acquisition.bde 23](#_Toc322951092)

[5.13.1.2 FIR180\_Top\_Level\_0207.bde 23](#_Toc322951093)

[5.13.2 Software 23](#_Toc322951094)

[5.13.2.1 Utils.h/.cpp 23](#_Toc322951095)

[5.13.2.2 ROIC\_IRC\_defines.h 23](#_Toc322951096)

[5.13.2.3 EEPROM.h/.cpp 24](#_Toc322951097)

[5.13.2.4 GeniCam.h/.cpp 24](#_Toc322951098)

[5.13.2.5 GeniCamLib.h/.cpp 24](#_Toc322951099)

[5.13.2.6 IRC\_controller.cpp 24](#_Toc322951100)

# Introduction

## Définition du document

Ce document présente les changements qui ont été réalisés pour effectuer l’implantation de la roue à filtres spectraux. Ces modifications ont été faites principalement pour le détecteur Hercules mais les modifications ont aussi été apportées au détecteur ISC0207 pour fin de test avec l’E-grade disponible.

## Documents de référence

* [SFW PDR](file:///\\thomson\disqueTELOPS\Projets\IRCDEV\Technique\Electronique\M9%20-%20ROIC\SFW\SFW%20PDR.docx)

## Acronymes

PDR Preliminary Design Review

DDR Detailed Design Review

GC GeniCam

FPA Focal-plane array

PPC Power PC

FPGA Field-programmable gate array

SFW Spectral Filter Wheel

AEC Automatic Exposure Compensation

EHDRI Enhanced High Dynamic Range Imaging

AGC Automatic Gain Control

RPM Revolutions per minute

# Implantation

## Paramètres des headers

Les paramètres suivants ont été ajoutés aux headers des images :

* SFWPosition (# du filtre)
* SFWEncoderAtExposureStart
* SFWEncoderAtExposureEnd

Les paramètres suivants ont besoin d’être ajoutés aux headers des images :

* SFWMode
* SFWSpeed
* SFWSpeedSetPoint

### SFWPosition

Contient le numéro du filtre lors de l’acquisition de l’image. Si le filtre n’est pas valide durant tout le temps d’intégration alors sa valeur est mise à invalide (255).

### SFWEncoderAtExposureStart

Contient la position de l’encoder lors du début de l’acquisition.

### SFWEncoderAtExposureEnd

Contient la position de l’encoder lors de la de l’acquisition.

## SFWPosition

Le registre GC SFWPosition permet de lire la position courante du filtre. Il ne faut pas confondre ce registre GC avec le paramètre du header du même nom. La valeur 255 indique un filtre invalide. Pour qu’on filtre soit valide, il faut que la SFW soit « locked » c.-à-d. que sont « homing » soit fait et que la position du filtre se trouve à l’intérieur de la plage de validité d’un filtre.

La plage de validité d’un filtre est mis à jour par :

[SFW\_UpdateFilterRanges(deltaTheta1, deltaTheta2)](file:///D:\Telops\FIR-00180-IRC\src\SFW\driver\SFW_Processing.cpp)

Les angles passés en argument proviennent de la fonction :

[CalculateMaximalValues(\*pGCRegs, changedParameter)](file:///D:\Telops\FIR-00180-IRC\src\SFW\driver\SFW_MathematicalModel.cpp)

Cette fonction doit être appelée à chaque fois que la taille de fenêtre est modifiée. La taille de fenêtre est le seul paramètre modifiable qui influence la plage de validité des filtres. CalculateMaximalValues() doit aussi être appelée lorsque le temps d’exposition et le frame rate sont modifiés mais seulement lorsque le mode SFW est SFW\_ SynchronouslyRotating.

Pour ne pas que CalculateMaximalValues prennent trop de temps à s’exécuter, le paramètre « changedParameter » indique quelle est la modification et effectue seulement les parties de l’algorithme que cela modifie. Plusieurs variables globales permettent de mémoriser des valeurs de l’algorithme qui sont réutilisables sans effectuer de nouveau calcul. Le même algorithme que celui conçu dans MATLAB a été implanté et donne les mêmes résultats dans les mêmes conditions.

## SFWPositionSetpoint

Lors de la modification de ce registre GC, la fonction suivante est appelée :

[SFW\_ChangeModeRequest(\*GCRegs)](file:///D:\Telops\FIR-00180-IRC\src\SFW\driver\SFW_Processing.cpp)

Si le « homing » du moteur n’est pas fait, il fera son homing pour connaître sa position absolue. Si le « homing » de la SFW n’est pas encore fait, le moteur continue à tourner pour atteindre l’index de l’encodeur de la SFW. Ensuite l’offset entre les deux encodeurs est enregistré pour les futures conversions entre les positions de la SFW et les positions du moteur :

La position du moteur convertie en « counts » de la SFW est soustraite à la position de la SFW pour calculer l’offset entre les deux. Ensuite on peut connaître le déplacement du moteur nécessaire à atteindre une position de la SFW :

Lorsque la roue passe du filtre 7 au filtre 0, la roue revient sur ses pas et effectue un rotation de 7/8 de tour dans le sens inverse.

## SFWSpeedSetpoint

En mode SynchronouslyRotating, SFWSpeedSetpoint est calculé automatiquement à partir de la valeur du frame rate :

La nouvelle valeur du frame rate est immédiatement recalculée pour correspondre à la valeur arrondie du RPM :

En mode AsynchronouslyRotating, SFWSpeedSetpoint et AcquisitionFrameRate sont découplée et SFWSpeedSetpoint se fait assigner par accès GC.

La vitesse envoyé au moteur est SFWSpeedSetpoint \* GEARS\_RATIO pour que la roue tourne à la vitesse désirée.

## ExposureTime

Les registres GC ExposureTime, EHDRIExposureSelector, EHDRIExposureTime, EHDRIExposureOccurence ne sont plus utilisés car remplacés par ExposureTimeSFW\_EHDRI\_ et EHDRIExposureOccurrence\_.

### SFWMode

Lorsque la roue est en mouvement les temps d’exposition pour le FPA Hercules la position de la roue doit être envoyée une acquisition en avance. Un générique « FPA\_EXP\_DELAY » a donc été ajouté à Image\_Sequencer pour pouvoir indiquer ce comportement. NEXT\_FILTER est donc utilisé au lieu de CURRENT\_FILTER lorsque FPA\_EXP\_DELAY vaut 1. Pour le FPA Hercules, ce délai est nécessaire à la transmission des bons temps d’exposition, SFWPosition et EHDRI au header.

À cause de l’ajout de ce délai et car on peut changer le temps d’exposition en temps réel lorsque le mode EHDRI n’est pas activé, il faut pouvoir se souvenir des 3 derniers temps d’exposition en cas de modification. On utilise donc l’espace non-utilisé par l’EHDRI dans le tableau des 32 temps d’exposition pour enregistrer les valeurs et on boucle sur l’index d’EHDRI du tableau lors d’une nouvelle écriture de temps d’exposition.

Pouvoir écrire de nouveaux temps d’exposition pendant une acquisition implique une modification du code pour le FPA Hercules. La mémoire utilisée doit maintenant être dual port pour pouvoir être écrite en même temps qu’être lue.

### EHDRIMode

Le mode EHDRI est disponible lorsque la SFW est en mode « fixed » ou « rotating ». Les tableaux ont été modifiés pour contenir les 32 valeurs possibles. En mode « rotating » la ram contenant la séquence d’EHDRI est générée pour seulement 128 valeurs et le résultat est répété 8 fois pour remplir la mémoire. En mode « fixed » la mémoire est remplie avec 1024 valeurs comme sans SFW.

## Trigger

En mode SynchronouslyRotating, les configurations du trigger sont contournées pour que les trigs proviennent uniquement du bloc SFW lorsqu’un filtre devient valide.

## AEC

L’AEC a été modifié pour enregistrer le numéro du filtre. Lors de l’appelle de la fonction de mise à jour du temps d’exposition, le numéro du filtre est lu et c’est seulement le temps d’exposition correspondant qui est modifié. Pour connaître le facteur de converge du temps d’exposition en fonction de la constante de temps, le timestamp contenu dans le header est toujours utilisé. Ce facteur sera toujours bon peu importe l’instabilité de la roue ou des autres trigs.

## Vidéo

### NTSC

Pour l’instant, en mode « fixed » toutes les images sortent en NTSC. Lorsque l’EHDRI ou SFW\_ROTATING sont actifs, il faut choisir quel numéro de filtre et quel numéro d’EHDRI peut sortir par NTSC. Il n’y a aucune vérification sur la calibration qui se trouve en mémoire pour l’instant. Lors de la détection d’un mismatch, il faudra désactiver l’affichage de la température et mettre l’AGC en mode automatique pour que ne qu’il utilise des températures comme paramètres d’affichage.

### AGC

Aucune modification nécessaire à l’algorithme de l’AGC puisque l’image est bloqué en amont du bloque AGC. Ce bloc ne voit donc qu’un seul temps d’exposition. Il faudra simplement le mettre en mode automatique lors de la détection d’un mismatch de calibration.

## Expansion board

Les signaux de l’encodeur sont routés correctement à travers le board d’expansion jusqu’au ROIC. La communication RS232 avec le contrôleur du moteur se fait aussi correctement.

## EEPROM

Ajout des variables suivantes qui contiennent des informations sur la SFW :

* uint16\_t gEEPROMMaxFWSpeed;
* uint16\_t gEEPROMNbPixelXMax;
* uint16\_t gEEPROMNbPixelYMax;
* uint16\_t gEEPROM\_SFW\_NbEncoderCyclesPerTurn;
* float gEEPROM\_SFW\_OpticalAxisPosX;
* float gEEPROM\_SFW\_OpticalAxisPosY; // Inches
* float gEEPROM\_SFW\_MountingHoleRadius; // Inches
* float gEEPROM\_SFW\_BeamMargin; // Inches
* float gEEPROM\_SFW\_cornerPixDistX; // Inches
* float gEEPROM\_SFW\_cornerPixDistY; // Inches
* float gEEPROM\_SFW\_centerPixRadius; // Inches
* float gEEPROM\_SFW\_cornerPixRadius; // Inches

Les modifications dans EEPROM.cpp, l’assignation d’adresse mémoire et l’écriture dans l’EEPROM doivent être faites.

## SFW Feedback

Les blocs FPGA ont été conçus et permettre d’avoir accès à la position, la vitesse, l’état (locked) et les erreurs de la SFW.

La seule erreur détectée est une vitesse plus grande que celle maximale. Il y a une bonne marge de manœuvre pour ne pas que la roue s’arrête inutilement. (ex. : 7000 RPM pour un vitesse max. de 6000 RPM). Cette valeur provient par accès WB de l’EEPROM.

Les signaux de sortie sont le numéro de filtre, le prochain numéro de filtre, un trig quand un filtre devient valide. Le bloc sort aussi les rising et falling positions, et le numéro de filtre en fonction du signal d’intégration qui entre dans le bloc.

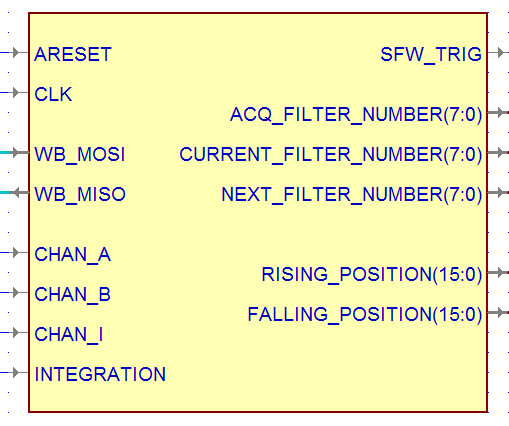


Figure 1 : entité SFW

Le signal d’intégration étant nécessaire pour SFWPosition, SFWEncoderAtExposureStart et SFWEncoderAtExposureEnd, il a fallu en créer un pour le FPA Hercules. Les autres FPA possédaient déjà leur signal d’intégration. Il débute lors de la réception du trig et se termine après 100MHz/80MHz\*CountsEnClkduFPA. Cette conversion est nécessaire puisque l’horloge interne est à 100Mhz et on possède le temps d’intégration en fonction de l’horloge du FPA.

Dans le cas du FPA Hercules il faut aussi ajouter un délai sur l’EHDRI et le # du filtre entrant pour que ceux sortant correspondent avec l’image dans le bus LocalLink. Lors d’une acquisition, la première image est rejetée automatiquement puisque son EHDRI et son # de filtre ne corresponde pas avec le bon temps d’exposition.

## SFW Processing

Le code de SFW Processing permet de gérer les transitions de la roue à filtre mais aussi à détecter les erreurs possibles. Lors d’une erreur la SFW arrête ses commandes mais pas lors d’un warning.

Si le homing de la SFW ne réussit pas à se faire après 4 tours du moteur + le déplacement du moteur lors de son homing, alors on lève une erreur. Cela peut détecter une roue à filtre manquante, une courroie manquante et un défaut de l’encodeur. On a besoin de deux tours de SFW car on doit trouver l’index et ensuite trouver le filtre initial.

Lorsque la position demandée sort de la plage de validité du filtre ou que la vitesse dépasse les marges d’erreur identifiées préalablement, un warning est envoyé à l’utilisateur lui indiquant que la roue est instable. Cela peut détecter une courroie défectueuse, un mauvais asservissement ou une perturbation externe sur la SFW.

Lors d’une mauvaise communication avec le contrôleur, une erreur est levée. Lors d’un trop long délai du comportement de la SFW ou de la communication avec le contrôleur, une erreur de SFW timeout est levée. Cela permet de détecter une trop grande friction, un mauvais asservissement et une perte de communication avec le contrôleur.

Plusieurs warnings peuvent être levées leur d’une erreur de communication temporaire avec le contrôleur du moteur ou d’un trop long délai sporadique avant d’atteindre une cible.

Lorsque le contrôleur du moteur tombe en erreur (Courant trop élevé, …), l’erreur est détectée et une erreur GenICam est levée. Cela permet de détecter toute défaillance du contrôleur, un blocage ou une trop grande friction de la SFW. Une autre solution sera de regarder le courant utilisé en fonction du temps pour détecter un trop gros effort du moteur. Par contre, le signal de courant n’est pas facilement accessible.

## Contrôleur du moteur

Le contrôleur du moteur ressemble fortement à celui du projet RFILW. La communication RS232 d’IRCDEV fonctionne par polling au lieu de par interruption.

Ajout de RS232\_Maxon\_LowPriorityTasks() pour détecter les erreurs de RS232 dans la communication avec le contrôleur du moteur. Cela génèrera des warnings GenICam lorsque les buffers de lecture ou d’écrire sont pleins.

Ajout d’états pour positionMode\_t et velocityMode\_t. Ces nouveaux états servent à vérifier le status du contrôleur du moteur suite à un enable. Cette modification a été nécessaire puisque tant que l’enable n’est pas terminer le contrôleur ignore les commandes envoyées.

Ajout des vérifications d’erreurs du contrôleur. Lorsqu’une erreur est détectée, le contrôleur passe en mode FW\_ERROR\_MODE avant de retomber en mode IDLE.

# Implantation à faire

## Paramètres du header

Ajouter les paramètres manquants :

* SFWMode
* SFWSpeed
* SFWSpeedSetPoint

## EEPROM

Assigner les adresses et ajouter les nouvelles variables à l’EEPROM. Modifier EEPROM.cpp pour aller lire ces valeurs et les mettre dans les variables.

(Voir : 2.11 EEPROM)

## AGC

Mettre l’AGC en mode automatique lorsqu’il y a un mismatch entre la calibration et la sortie vidéo. Lors d’un mismatch, il ne faut pas que la température apparaisse sur l’image vidéo.

## Calibration

Permettre d’avoir plusieurs sets de calibration, choisir lesquels appliquer et à quel moment.

## EHDRI

Modifier HDRA\_UpdateExpectedExposure() pour pouvoir estimer correctement pour chacun des filtres les bon temps d’exposition nécessaire en fonction de leur calibration respective.

## AEC+

L’AEC+ et ce que cela implique pour la SFW se retrouvera dans le PDR de l’AEC+.

## État de la caméra

Vérifier l’état de la roue avant de partir une acquisition. Devant le bon filtre en mode fixe et à la bonne vitesse en mode rotating.

## gEEPROMSFWPresent

Ajout de ce flag pour savoir si une SFW est fixée à la caméra. S’assurer que toutes les fonctions reliées à la SFW ne sont pas appelées et que les fonctions qui accèdent aux numéros de filtre et autres variables fonctionnent correctement sans ces valeurs ou avec leurs valeurs par défaut.

## Autres détecteurs (plus tard)

Le FPA Hercules et ISC0207 ont été modifiés pour être compatible avec la SFW. Les autres détecteurs devront être modifiés lorsqu’un détecteur de leur type devra être mis dans une caméra avec SFW.

# TESTS

## Faits

### Contrôle de la roue à filtre

Avec l’accès aux registres GeniCam, le board ROIC peux contrôler la roue en position et en vitesse.

Lors de la configuration du contrôleur de la SFW, les commandes ont dû être adoucies pour que ne pas le moteur soit instable (n’arrive pas à atteindre exactement sa consigne et fait du bruit car il change constamment de direction). Avec la roue et les paramètres actuels ont a environ ±15% d’erreur pour le RPM à environ 100 RPM. En basse vitesse (10 RPM), les erreurs sont élevées, i.e. le RPM entre 3 et 17. L’échantillonnage de la vitesse se fait très rapidement (60/4096 = 14.6 ms). Ce temps pourrait être allongé ce qui moyennerait les fluctuations du RPM lu.

Lorsqu’asservi en position, des erreurs de ±5 counts (±0.5°) sont perceptibles. Comme la position du filtre se situe en ce centre cette erreur ne cause pas de problème.

De nouveaux tests et une nouvelle configuration du contrôleur seront nécessaires avec le montage final. Une vérification de toute la plage de vitesse en RPM devra être faite pour connaître les valeurs possibles de vitesse. Cette identification servira à connaître le comportement de l’instabilité de vitesse pour chaque valeur de RPM demandée. Cette courbe servira à ajouter des marges de sécurité sur le frame rate demandé lors du calcul des valeurs maximales.

### Synchronisation des encodeurs

Suite à une consigne de la SFW en position, une synchronisation des encodeurs doit être faite pour connaître quelle position demandée au moteur pour que la SFW atteigne la position désirée.

Toutes les combinaisons possibles des états initiaux (encodeur moteur (non-)valide, encoder roue (non-)valide, états de la SFW (fixed/rotating), position initiale de la roue) ont été testées. La position atteinte est toujours bonne suite à la synchronisation avec ±5 counts.

### Écriture des headers

Les nouveaux paramètres suivants ont été vérifiés suite à des acquisitions en mode fixe et rotating et leurs valeurs sont correctement mises à jour : SFWPosition, SFWEncoderAtExposureStart et SFWEncoderAtExposureEnd.

Les paramètres suivants devront être vérifiés suites à leur ajout dans le firmware : SFWMode, SFWSpeed, SFWSpeedSetPoint

### Génération du trigger en synchronously rotating

Pour le FPA ISC0207, lorsque la caméra est en mode synchronously rotating les trigs proviennent des débuts des filtres. Cela a été confirmé en regardant le paramètre SFWEncoderAtExposureStart suite à des acquisitions. Ce paramètre contenait la valeur de début du filtre correspondant au numéro dans SFWPosition.

### EHDRI et SFW

Les temps d’exposition demandés au FPA, et leur séquence, sont toujours bons avec toutes les combinaisons possibles de l’EHDRI et de la SFW.

## À faire

### Génération du signal d’intégration (Hercules)

Comme ce signal d’intégration a dû être développé pour le FPA Hercules, des tests devront être faits pour s’assurer de son bon fonctionnement dans tous les modes possibles modifiants le temps d’intégration.

### Synchronisation du filtre

(Surtout pour la Hercule) En rotation, testez si le numéro du filtre attendu est bel et bien le filtre de l’image acquise. Cette vérification est nécessaire due au délai entre l’envoi du temps d’exposition et la réception des données de l’image. Un filtre unique ou un obturateur pourra être utilisé pour clairement identifier une position parmi toutes les autres.

### Mises à jour temps réel

Vérifier le bon fonctionnement de la modification des temps d’exposition lorsqu’une acquisition est en cours. Il ne faut pas que le temps d’exposition dans le header soit écrasé par la nouvelle valeur si l’ancienne est celle utilisé dans l’image en cours.

Ce test est surtout pour la Hercules car modification majeure du bloc FPA dans le FPGA.

### AEC

Tester l’AEC dans toutes les configurations possibles pour vérifier son bon fonctionnement. (EA\_Once, EA\_Continuous) (Fixed, SynchronouslyRotating)

### Centre des filtres

Élaboration de tests pour connaître la position des centres de filtre.

### Répétabilité du positionnement

Analyser toutes les acquisitions pour savoir si l’atteinte du centre des filtres est répétable de fois en fois et avec son évolution dans le temps.

### Stabilité de la SFW

Vérifier la stabilité de la SFW avec le montage final. Est-ce que la roue peut reculer d’un count lorsqu’elle tourne lentement ? Si oui, il faudra ajouter une hystérésis dans le bloc FPGA pour que l’acquisition reste valide tant que le recul reste dans la marge d’erreur.

Vérifier les marges d’erreur des filtres en fonction des vitesses possibles. Ces marges seront utilisées dans le code pour afficher des warnings à l’utilisateur si ces marges d’erreurs sont dépassées.

### Global

Test dans différentes configurations du système global final pour tous les détecteurs désirés. Un plan de test global devra être conçu.

# Modification du firmware

## SFW

### Source

#### fw\_decoder.vhd

Nouveau fichier, permet de décoder un signal de quadrature. On peut fixer à l’aide d’un generic : CLOCK\_FREQUENCY, ZERO\_PHASE et SPEED\_PRECISION\_BIT.

On peut par WB mettre le nombre de counts de l’encodeur dans la variable NB\_ENCODER\_COUNTS. Le bloc FPGA peut donc être dynamique par rapport à la valeur inscrite dans l’EEPROM, i.e. sans besoin d’être recompilé.

Ce module a été testé un l’aide d’un simulateur d’encodeur de quadrature. Différentes vitesses, directions, NB\_ENCODER\_COUNTS, CLOCK\_FREQUENCY et ZERO\_PHASE ont été simulés. Tous les tests se sont avérés concluant.

Un facteur de précision a été ajouté à la vitesse mesurée SPEED\_PRECISION\_BIT pour pouvoir avoir des décimal sur le vitesse en RPM. Cela diminue le refresh rate par le même facteur.

#### SFW\_WatchDog.vhd

Ce fichier n’a pas été créé et ses fonctionnalités ont été transférées à SFW\_Ctrl.vhd

#### SFW\_Ctrl.vhd

Permet de connaître le filtre acquis lors de la montée et de la descente du signal d’intégration. Il permet de sortir la position de la roue de début et de fin d’intégration. Permet aussi de sortir le filtre courant et le prochain. Lorsque la roue est fixe, le filtre courant et le prochain sont forcés à la valeur du filtre demandé.

Ce fichier a été testé avec plusieurs vitesses de roue, à plusieurs temps d’intégration différents et fonctionne correctement.

Il vérifie que la roue tourne plus lentement que ça vitesse maximale, spécifiée par accès WB.

#### SFW.bde

Entité maîtresse de SFW

### Driver

#### FW\_timer.h/.cpp

Ces fonctions utilisent maintenant GETTIME() et elapsed\_time\_us(). Elles utilisent maintenant un tableau pour les différents timers au lieu d’une fonction différente par chaque timer.

Ajout du timer FW\_TIMER\_VERIFY\_ERRORS qui sert à vérifier périodiquement si le contrôleur est en erreur et ce même si il a atteint sa cible. Cela permet de détecter si le contrôleur a arrêté de fonctionner.

#### maxonProcessing.h/.cpp

Modification des timers pour être conforme avec FW\_timer. Modification des gestions des erreurs, maintenant il envoie ses erreurs à FWController .

Modification de MaxonGetResponse() pour qu’il retourne correctement la longueur du message de réponse.

#### RS232\_Maxon.h/.cpp

Maintenant la fonction fonctionne par polling au lieu de par interruption. Elle est basée sur l’UART du GPS. Elle doit donc être appeler prériodiquement par la boucle while du main.

Ajout de RS232\_Maxon\_LowPriorityTasks() qui sert à vérifier les erreurs de buffers pour la transmission RS232

#### EPOS\_Control.h/.cpp

Aucune modification pour l’instant

#### maxonUtilities.h/.cpp

Modification de SetProfilePositionMode(), SetProfileVelocityMode() et SetHomingMode() pour qu’ils n’envoient le Start. C’est d’autres fonctions qui vont envoyer le Start lorsque le status du enable sera bon.

Ajout de la fonction SetStartHoming() pour faire le homing quand le contrôleur aura fait son enable.

Modification de GetResponse() pour qu’il retourne correctement la longueur du message de réponse.

#### FWController.h/.cpp

Supporte la nouvelle gestion des erreurs et des timers pour SFW.

Ajout de la vérification, si Homing est valide.

Ajout d’états pour positionMode\_t et velocityMode\_t. Ces nouveaux états servent à vérifier le status du controlleur du moteur suite à un enable. Cette modification a été nécessaire puisque tant que l’enable n’est pas terminer le controlleur ignore les commandes envoyées.

Ajout de la vérification des bits d’erreur du contrôleur du moteur. Cette vérification continue à se faire périodiquement même lorsque le contrôleur à atteint sa cible. Cela permet de détecter des erreurs survenues après l’atteinte de la cible. Cela générera une erreur GenICam.

Ajout de la vérification périodique des erreurs du contrôleur EPOS2.

#### SFW\_Processing.h/.cpp

Création de ces fichiers. SFW\_Processing() permet de changer entre les différents SFWMode. Si le mode position est choisi, il faut que le Homing soit valide sinon on recalcule l’offset entre les deux encodeurs à forçant le contrôleur du moteur à la position 0 et en lisant la position de l’encodeur de la roue. Si la roue n’est pas « locked » alors on tourne la roue jusqu’à temps qu’elle devienne « locked ».

SFW\_Processing permet de convertir les erreurs de FWController en erreurs GenICam avec update\_GC\_error\_reg().

La fonction SFW\_UpdateFilterRanges() permet de convertir les angles disponibles pour les filtres spectraux en plage de validité des filtre exprimée en position d’encodeur.

La fonction SFW\_GetFilterPosition() permet de déterminer à quel filtre est associé la position courante de la roue à filtre en fonction des plages de validité des filtres.

#### SFW\_MathematicalModel.h/.cpp

Création de ces fichiers. CalculateMaximalValues() permet de calculer les valeurs de registres GeniCam maximales lorsque la roue à filtre tourne. InitMathematicalModel() doit être appelé une fois au démarrage car les fonctions interne ne mettre à jour que les registres nécessaire lorsque la fonction CalculateMaximalValues() est appelée. Le paramètre « changedParameter » permet de spécifier quelle variable a changé de valeur depuis le dernier appelle. Cela est fait pour éviter des calculs inutiles. Les valeurs sont mises à jour dans des variables internes. Il faut ensuite les lire avec les fonctions SFW\_GetXXXXXXXXXX().

CalculateMaximalValues() doit être appelée même en mode fixe lorsque la taille de la fenêtre change pour mettre les valeurs limites des filtres en fonction de l’encodeur de la roue à filtre. Pour se faire la fonction SFW\_UpdateFilterRanges() est appelée à l’intérieur de CalculateMaximalValues().

### Testbench

#### quadrature\_encoder.vhd

Ajout d’un module pouvant simuler l’encodeur en quadrature. Génère les 3 signaux en fonction de la vitesse demandée, la position du zéro et du nombre de count sur l’encodeur.

## TMI\_LL\_Histogram

### Source

#### Histogram\_mcw

Modification des fichiers Simulink et recompilation pour associer un numéro du filtre spectral aux histogrammes. Ajout des ports ext\_data\_in2(31:0) et ext\_data\_out2(31:0).

#### ll\_tmi\_histogram\_1024.bde

Ajout des ports ext\_data\_in2(31:0) et ext\_data\_out2(31:0).

## AEC

### Source

#### AEC\_Ctrl.vhd

Ajout de la gestion du numéro de filtre : Latch et lecture par les registres GenICam.

#### AEC.bde

Ajout du port entrant FILTER\_NB et la gestion interne du numéro de filtre.

### Driver

#### AEC.h/.cpp

AEC\_ExpTime est maintenant un tableau pour contenir la valeur de l’AEC des 8 filtres. À chaque itération, le numéro du filtre est lu à partir des registres GenICam et cela permet de mettre à jour le bon index de AEC\_ExpTime.

AEC\_LastHistTimeStamp est devenu un tableau contenant le timestamp pour chacun des filtres. Quand AEC\_Rate\_Mode est manuel, le taux est 8 fois plus rapide si la roue tourne de façon synchrone pour avoir la même constante de temps. En mode automatique, le timestamp au bon index est lu ce qui garantit la bonne constante de temps pour chaque filtre.

Ajout de AEC\_OnceDone pour vérifier que les 8 temps d’exposition ont convergé avec de terminer l’AEC. Quand le mode est à Once, on force l’AEC\_FilterRate à 1.

Afin de calculer les paramètres en fonction du temps de réponse demandé, le timestamp est toujours utilisé. Cela permet d’avoir toujours la bonne valeur malgré une instabilité du trig.

## Image Sequencer

### Source

#### IRC\_define.vhd (Common\_HDL)

Modification de “exp\_time\_array\_t” pour avoir une longueur de tableau de 32.

#### img\_sequencer\_Ctrl.vhd

Ajout de CURRENT\_FILTER\_NUMBER en entré pour envoyer au FPA le bon temps d’exposition.

Ajout de SFW\_EXP\_IDX pour sortir le numéro du filtre

Ajout des 32 registres WB pour pouvoir écrire les temps d’expositions pour toutes les combinaisons de filtres et d’EHDRI. En mode de mise à jour temps réel, les index d’EHDRI sont utilisés comme buffer circulaire.

#### img\_exp\_time\_sm.vhd

Supporte maintenant SFW\_FILT\_IDX\_IN/OUT.

#### Image\_Sequencer.bde

Prend maintenant en entrée CURRENT\_FITLER\_NB, NEXT\_FILTER\_NB, ACQ\_FILTER\_NB et le generic FPA\_EXP\_DELAY. FPA\_EXP\_DELAY doit être fixé à 1 pour le détecteur Hercules. Cela choisira l’index de next\_filter\_nb au lieu de current\_filter\_nb. Les positions de début et de fin d’intégration sont aussi ajoutées en entrée.

### Driver

#### Image\_Sequencer.h/.cpp

Supporte maintenant les 32 valeurs possibles de temps d’exposition. Met à jour le régistre WB : A\_NEW\_EXP\_TIME\_FILT quand une modification du temps d’exposition est faite.

ImageSequencer\_ChangeExposure() prend maintenant comme argument le numéro du filtre qu’elle désire mettre à jour.

## FPA (isc0207)

### Source

#### fpa\_intf\_cfg\_receiver.vhd

Ajout de la gestion des 32 temps d’exposition par WB et modification des index pour supporter la position de SFW. Maintenant lors d’un changement de temps d’exposition real-time, on boucle dans le tableau en changeant l’index EHDRI (utilisez comme tampon pour les temps d’exposition précédents).

#### fpa\_isc0207\_interface.bde

Modification des index pour supporter la position de SFW.

### Driver

#### fpa\_intf.h/.cpp

Ajout de la gestion des 32 temps d’exposition.

Pour FPA\_ChangeExposure(), il faut maintenant spécifier le numéro du filtre à mettre à jour.

## FPA (Hercules)

### Source

#### Hercules\_COM\_Register.vhd

Ajout du port NEW\_EXP\_TIME\_FILT et suppression de FPA\_EXP\_IDX\_OUT et INT\_TIME\_SET. C’est maintenant la tâche de Hercules\_exp\_time.vhd de sortir FPA\_EXP\_IDX\_OUT et l’équivalent de INT\_TIME\_SET.

Ajout de la gestion des 32 temps d’intégration.

Maintenant lors d’un changement de temps d’exposition real-time, on boucle dans le tableau en changeant l’index EHDRI (utilisez comme tampon pour les temps d’exposition précédents).

Séparation du signal LOAD du registre de contrôle car on peut vouloir mettre à jour le tableau durant que ce registre est en mode START.

Ajout du signal WR\_HDRI\_INDEX qui indique quel index EHDRI du tableau de temps d’exposition qu’il faut mettre à jour. Sa valeur est déterminée par le registre WB lorsque l’acquisition n’est pas démarrée. Si en mode acquisition alors WR\_HDRI\_INDEX prend la prochaine valeur par rapport à l’index EHDRI courant.

#### Hercules\_HDRI\_Ram.vhd

La mémoire est augmentée pour pouvoir contenir les 32 valeurs de temps d’exposition. La mémoire est aussi devenue une dual port ram pour pouvoir écrire et lire en même temps lors du changement des temps d’exposition en temps réel. Les adresses mémoire ont été réarrangées pour faciliter l’accès direct aux temps d’exposition : SFW\_FILT\_IDX\_IN & HDRI\_INDEX & bytes.

Ajout du port WR\_HDRI\_INDEX pour indiquer à quelle adresse EHDRI on désire mettre les valeurs à jour. Cette valeur provient de Hercules\_COM\_Register.

#### Hercules\_Header\_Extractor.vhd

CURRENT\_EXP\_TIME remplace maintenant INT\_TIME\_SET. EXP\_TIME\_ARR et FPA\_EXP\_IDX\_OUT ont été enlevés. Ce n’est plus la responsabilité de header\_extractor de sortir l’index du temps d’exposition. Il ne fait que recevoir le temps d’exposition attendu de Hercules\_exp\_time et vérifie qu’il concorde bien avec celui extrait du header. S’ils ne concordent pas, FPA\_ERR est activé.

#### Hercules\_trig\_proc.vhd

Ajout d’un délai de 1 acquisition sur le signal ready pour que le bon temps d’exposition se rendre au FPA en EHDRI ou en SFW\_Rotating. Cela est dû au délai de 1 du FPA Hercules.

#### Trigger\_Filter.vhd

Retrait du port Ready puisqu’il était déjà géré dans le composant supérieur. Ajout de la détection des trigs seulement lors du rising edge.

#### Hercules\_exp\_time.vhd

Ajout de ce nouveau composant. Il sert à gérer la latence des index du temps d’exposition pour sortir correctement les valeurs de FPA\_EXP\_IDX\_OUT et de SFW\_FILT\_IDX\_OUT. Aussi, un signal AQC\_INT de bonne durée est généré à partir de la table de temps d’exposition. Comme cette table est en cycle d’horloge pour le FPA (80 MHz), cette valeur est multipliée par 5/4 pour atteindre la bonne valeur de compteur pour l’horloge utilisée (100 MHz).

La valeur du temps d’exposition attendue est aussi sortie pour son utilisation par d’autres composants.

#### Hercules\_COM\_Controller.bde

Ajout du port new\_exp\_time\_filt.

#### Hercules\_COM\_Manager.bde

Ajout du port new\_exp\_time\_filt et sfw\_filt\_idx\_in.

#### Hercules\_interface.bde

Ajout de Hercules\_exp\_time et réarrangement des connexions inter-composants. Ajout des ports SFW\_FILT\_IDX\_IN, SFW\_FILT\_IDX\_OUT, NEW\_EXP\_IDX\_IN.

### Driver

#### fpa\_intf.h/.cpp

Modification d’adresse WB.

Ajout de FPA\_SetEHDRISel() après un FPA\_Start(). Sans l’appelle de cette fonction, le FPA ne se mettait jamais en mode EHDRI car le FPA\_Stop() appeler au début de FPA\_Start() reset son mode. Le FPA lit ses valeurs en mémoire si EHDRI est actif ou si la roue à filtre tourne.

Maintenant envoie les 32 temps d’exposition au FPGA.

Ajout de la fonction FPA\_SetWrHdriIndex(const t\_FpaIntf \*a, uint32\_t index). Elle sert à envoyer au FPGA le numéro d’EHDRI que le PPC désire mettre à jour dans le tableau des temps d’exposition pour la prochaine séquence de 8 valeurs que le PPC va envoyer.

#### Hercules\_Processing.h/.cpp

Maintenant, la fonction Hercules\_IntegrationTimeCommand() peut envoyer:

* Soit les 32 temps d’exposition à la mémoire du FPGA si en mode EHDRI ou SFW\_Rotating,
* Soit les 8 nouveaux temps d’exposition si en mode sans EHDRI et en mode SFW\_Rotating ou SFW\_Fixed,
* Soit le temps d’exposition désiré en SFW\_Fixed sans EHDRI.

## FRAME\_BUFFEr\_COADD

### Source

#### WB\_CoAdd\_CTRL.vhd

Certaines adresses WB ont été changées et les ports suivant ont été ajoutés : VERIFY\_INDEX, FPA\_EXP\_VIDEO\_IDX\_OUT et SFW\_FILT\_VIDEO\_IDX\_OUT.

#### MAIN\_CTRL.vhd

Les ports suivants ont été ajoutés : VERIFY\_INDEX, FPA\_EXP\_IDX\_IN, SFW\_FILT\_IDX\_IN, FPA\_EXP\_VIDEO\_IDX\_IN, SFW\_FILT\_VIDEO\_IDX\_IN et VALID\_INDEX.

Le port VALID\_INDEX est actif quand l’index n’est pas vérifier ou quand l’index est vérifié et que l’index du temps d’exposition courant concorde avec FPA\_EXP\_VIDEO\_IDX\_IN et SFW\_FILT\_VIDEO\_IDX\_IN.

#### FBC\_CTRL.bde

Ajout des ports FPA\_EXP\_IDX\_IN, SFW\_FILT\_IDX\_IN et VALID\_INDEX. Ajout des nouvelles connexions entre WB\_CoAdd\_CTRL et MAIN\_CTRL.

#### Frame\_Buffer\_CoAdd.bde

Ajout des ports FPA\_EXP\_IDX\_IN et SFW\_FILT\_IDX\_IN.

Ajout d’un composant LL\_Hole16 en avant de XBAR avec son FALL relié à VALID\_INDEX. Lorsque l’index du flow de donnée sera considéré comme invalide, ce flow de donnée disparaîtra. LL\_Hole16 a été modifié pour pouvoir latcher la valeur de FALL seulement lorsqu’aucun flow de donnée n’est présent. Le generic SW\_ON\_EOF\_g doit être mis à true.

#### CoAdd\_FullSyst\_Top.bde

Ajout du port SFW\_FILT\_IDX\_IN.

### Driver

#### Frame\_Buffer\_CoAdd.h/.cpp

Modification d’adresses WB. Ajout des registres WB : VERIFY\_INDEX, EHDRI\_VIDEO\_INDEX et SFW\_VIDEO\_INDEX.

VERIFY\_INDEX indique que pour que le flow de donnée soit accepté dans le composant XBAR, il faut que les index du temps d’exposition concordent avec EHDRI\_VIDEO\_INDEX et SFW\_VIDEO\_INDEX. VERIFY\_INDEX est actif seulement en mode vidéo et que soit l’EHDRI ou soit la roue à filtre sont actifs.

## EHDRI

### Driver

#### EHDRI\_Acquisition.h/.cpp

Lorsque la roue tourne de manière synchrone, on génère une séquence 8 fois moins longue mais on répète chaque index dans la BRAM pour que chaque filtre suive la même séquence d’EHDRI.

Mise à jour de HDRA\_UpdateExpectedTemp(). Utilisation de MaxExposureTime() et MinExposureTime() pour ensuite calculer les valeurs des températures.

Mise à jour de HDRA\_UpdateExpectedExposure(). Initialisation des valeurs d’EHDRI des 8 filtres. Cela met les mêmes valeurs qu’auparavant mais pour les 8 filtres différents.

Le tableau de valeur dans le code a dû être déclaré « static », puisque la pile du PPC ne peut contenir que 1024 bytes, ce qui est inférieur aux tailles des tableaux utilisés.

## Trig Ctrl

### Source

#### Trigger\_ctler\_kernel.vhd

Quand le mode du trigger est SfwTrig\_st, on prend l’entrée SFW\_SYNC\_TRIG pour trigger.

#### fpa\_trigger\_define.vhd

Ajout du mode SFW\_TRIG

#### fpa\_trigger.bde

Ajout du port SFW\_SYNC\_TRIG

#### fpa\_trigger\_ctler.bde

Ajout du port SFW\_SYNC\_TRIG

### Driver

#### Trigger.h/.cpp

Si la roue à filtre tourne de manière synchrone, on overwrite le mode à SFW\_TRIG et l’activation à RisingEdge.

## Video

### Source

#### AGC\_Ctrl.vhd

Puisque ll\_tmi\_histogram\_1024 prend maintenant en entrée le numéro du filtre, AGC\_Ctrl à ce nouveau port d’entrée et peut être lu à partir du WB. Ce paramètre n’est pas utilisé car le flow de data est filtré en amont si plusieurs filtres spectraux ou EHDRI sont disponibles

#### AGC.bde

Ajout du numéro de filtre en entrée.

#### Video\_top.bde

Ajout du numéro de filtre en entrée.

## HDR Merge

### Driver

#### HDR\_Merge.h/.cpp

Ajout de la position du filtre en mode fixe pour aller lire la bonne valeur de temps d’exposition. Cet ajout est pour avoir les bon temps d’exposition en mode fixe, ne fonctionne pas lorsque la roue tourne.

## AGC

Aucune modification n’a été nécessaire contrairement aux indications du PDR, car la fonction qui calcul la valeur du gain se fie sur les timestamps des frames et non sur le framerate demandé.

## Top Level

### VHDL

#### ROIC\_Acquisition.bde

Ajout de SFW. Branchement du WB sur le bus #15. Ajout en entrée des 3 signaux venant de l’encodeur de quadrature.

Ajout des informations sur les filtres pour AEC, Image\_Sequencer, video\_top.

Ajout du SFW\_SYNC\_TRIG pour fpga\_trigger.

La position du filtre est maintenant envoyée au FPA et reçu.

Ajout d’un generic FPA\_EXP\_DELAY pour l’Image\_sequencer.

#### FIR180\_Top\_Level\_0207.bde

Ajout des entrées pour l’encodeur de quadrature.

La position du filtre est maintenant envoyée au FPA et reçu.

### Software

#### Utils.h/.cpp

Ajout de fonctions pour calculer le temps d’exposition maximal et minimal parmi les 32 choix en fonction du mode SFW, du mode EHDRI et du nombre de temps d’exposition de l’EHDRI.

#### ROIC\_IRC\_defines.h

Ajout de l’adresse WB, SFW\_WBADD à l’adresse 15.

#### EEPROM.h/.cpp

Ajout de gEEPROMFilterPositions contenant la position des filtres spectraux.

Ajout de 12 variables pour les calculs dans SFW\_MathematicalModel.

#### GeniCam.h/.cpp

Modification de EHDRIExposureTimeAry en float\* pExposureTimeSfwEhdriAry[32] et EHDRIExposureOccurenceAry en float\* pEHDRIExposureOccurenceAry[4]. Maintenant on pointe vers les adresses des registres GenICam. On ne répète pas l’information on l’accède juste plus facilement à l’aide des index du tableau.

#### GeniCamLib.h/.cpp

Séparation de GC\_WindowChanged en GC\_WindowWidthChanged et GC\_WindowHeightChanged.

Ajout de CalculateMaximalValues() lorsque les paramètres qui y sont reliés changent et que le mode de la roue est synchrone. Lorsque FrameRate change, calcul du SFWSpeedSetpoint.

Ajout des callbacks pour SFWMode, SFWSpeedSetpoint, SFWPositionSetpoint et SFWSpectralFilterNameAlias.

Ajout des callbacks GC\_ExposureTimeChanged{1,5,9,13,17,21,25,29}() pour pouvoir savoir et indiquer le temps d’exposition de quel filtre doit être mis à jour.

La fonction GC\_ExposureTimeChanged2 s’appelle maintenant GC\_ExposureTimeChanged\_Step2() et prend comme argument l’index du temps d’exposition. Cet index n’est utiliser que lorsque l’index correspond à un EHDRI1 (index%4==0) et c’est pour mettre à jour les temps d’expositions en real-time. Tous les autres index peuvent donc avoir le même index qui n’est pas un multiple de 4 ce qui permet de ne pas créer des callbacks inutilement.

Création de GC\_EHDRINumberOfExposuresCallback() car HDRA\_UpdateExpectedExposure() ne doit pas être appelé lors du changement du nombre de temps d’exposition du EHDRI lorsque le mode d’EHDRI (EHDRISettingsMode) est « Advanced ».

#### IRC\_controller.cpp

Ajout des fonctions AGC\_Processing(), SFW\_Processing(), FWControllerProcess(), MaxonProcessing(), RS232\_Maxon(), RS232\_Maxon\_LowPriorityTasks() dans la boucle while.

Ajout des fonctions InitMathematicalModel(), SFW\_Reset(), RS232\_Maxon\_Init() dans l’initialisation. Apelle de SFW\_ChangeModeRequest() pour que le mode de la roue concorde avec la valeur des registres GeniCam.