FSUIPC4 pour Utilisateurs Avancés (version 4.60 ; Mars 2010)

Peter DOWSON

(traduction : Philippe HANTZBERG)

Pour les changements apportés depuis la version précédente , reportez-vous je vous prie au document « History » .

Options dans le fichier FSUIPC4.INI

Dans une installation de FSUIPC4 utilisateur-enregistré , toutes les options intéressantes peuvent être contrôlées à travers la fenêtre Options and Settings obtenue en sélectionnant le menu Add-Ons , puis FSUIPC (ALT , D , puis F) C'est la méthode recommandée , qui permet des changements « en vol » . Les changements effectués sont enregistrés dans un fichier , de telle sorte qu'ils sont conservés pour le prochain rechargement en mémoire .

Presque toutes ces options sont enregistrées dans la section [General] de FSUIPC4.INI , qui est un fichier texte éditable initialement créé pour vous dans le répertoire Modules (ou , sous Windows Vista , dans le répertoire FSX , dans Documents) . Seuls les paramètres soulignés ne sont pas ajustables dans la fenêtre Settings (pour un Utilisateur enregistré) .

Options Fenêtre Message

A la différence de presque tous les autres paramètres , ils sont tous disponibles aux Utilisateurs non enregistrés.

ShowMultilineWindow : ce sera « Yes » si la boite à cocher de la première page de FSUIPC4 (About+Register) est cochée. Les messages multilignes sont dirigés vers une fenêtre translucide comme celle utilisée par l'ATC de FS .

SuppressMultilineFS : détermine quels messages multilignes sont acheminés vers FS pour ses fenêtres messages. Si l'option ci-dessus est « Yes » , alors ce réglage n'est pas pertinent .

SuppressSingleLine : empêche l'affichage de messages simple ligne dans la fenêtre messages de Windows (ou de AdvDisplay) . De tels messages sont abandonnés purement et simplement si cette option est sélectionnée .

Voyez aussi **WhiteMessages** et **AdvDisplayHotKey** , documentés tous les deux dans la section « Autres Options » .

Options Météo générale

ClearWeatherDynamics : FSX implémente un algorithme de changement météo , « weather dynamics » , qui permet à la météo de changer individuellement en fonction du temps à chaque station météo . FSUIPC4 essaiera , par défaut , de couper l'action dynamique lorsqu'il sera requis via l'interface New Weather ; ou automatiquement lorsque la météo est réglée via l' « advanced weather interface » de FS98 et FS2000. En fait , actuellement , il apparaît qu'il ne peut pas couper complètement , seulement le ralentir. L'option se règle dans la page **Miscelleanous** de FSUIPC et par son paramètre INI.

OwnWeatherChanges : est réglé sur « No » par défaut. Si c'est réglé sur « Yes » , ça autorise des filtres pour les Nuages , les Vents et la Visibilité à s'appliquer à la météo de FS . ça permet au wind smoothing et à la visibilité graduée (toujours expérimentale) d'être réglés . L'option est commandée par des boites à cocher sur les Onglets Winds , Visibility et Clouds .

WeatherReadFactor : FSUIPC4 lit la météo de l'avion (interpolée depuis 3 stations) , la météo de la station la plus proche , et la météo globale (ou par défaut) , toutes à intervalles réguliers – la météo de l'avion étant lue à intervalles déterminés par la valeur

« WeatherReadFactor » . D'ordinaire , ça règle directement l'intervalle minimum en demi-secondes , et par défaut à 2 secondes .

Depuis la version 4.2 , ce paramètre commandé également la fréquence des M.A.J. de la météo de FS lorsque l'option « allow changes to FS own weather » est réglée (voir ci-dessus OwnWeatherChanges) .
Il ne devrait pas y avoir besoin de le changer , mais si vous pensez qu'il surcharge trop votre système , vous pouvez essayer un intervalle plus long , jusqu'à 30 secondes . Inversement , si vous n'avez pas besoin de lire la météo ou d'écrire des activités dans FSUIPC4 , vous pouvez les stopper toutes en réglant ce paramètre sur 0 .

Dans les options on-line , ce paramètre est actuellement changeable dans l'onglet Winds .

WeatherRewriteSeconds=1 : spécifie le nombre de secondes qu'emploie SimConnect pour « mélanger » la météo réécrite avec la météo actuelle . Théoriquement , les changements devraient être plus doux avec ce délai , quoique à l'expérience on ne voie pas grande différence entre 1 et 30 secondes . Idéalement , pour wind smoothing et graduated visibility la valeur 0 serait la meilleure , vous pouvez l'essayer . mais vous serez avertis : ça créera alors beaucoup de saccades !

SmoothBySimTime=No : Si vous le réglez sur « yes » , ça basera tout le fonctionnement de weather smoothing sur le temps passé dans FS , à la place du temps système réel . L'avantage est que ça stoppe pendant que FS est dans les menus , ou que ça se met sur Pause , ou que ça tourne plus vite ou moins vite en fonction du taux de simulation de FS .
Pour utiliser cette méthode de smoothing , changez ce paramètre [General] :
Notez que le smoothing est toujours réinstallé quand vous chargez un nouveau Vol en mémoire , ou que vous déplacez la localisation de l'avion via le Menu , ou que vous changez le mode Météo (theme\user\rel , etc …)

Winds

MaxSurfaceWind : permet à l'apparence du vent d'être limitée à une vitesse de vent maximale spécifiée en nœuds . Cette fonctionnalité est déconnectée si la valeur assignée ici est 0 .

WindDiscardLevel : ce paramètre installe une vitesse de vent dont les entrées , à partir d'un programme de commande météo externe , utilisant l'interface FS98 (**pas** les Interfaces Advanced ou New Weather) sont ignorées . La valeur par défaut est 400 nœuds . Si un programme météo essaie d'installer une vitesse de vent supérieure , ça sera ignoré et le réglage de vitesse antérieur pour cette couche de vent est conservé (Ce paramètre est fourni spécialement pour éviter la survenue de problèmes avec des programmes qui utilisent des données corrompues téléchargées depuis Internet ou d'autres problèmes) . Régler ce paramètre à 0 désengage complètement ce contrôle .

WindLimitLevel : ce paramètre donne une limite à la vitesse du vent , qui puisse être acceptée par un programme météo externe , utilisant l'interface FS98 (**pas** les Interfaces Advanced ou New Weather) . La vitesse par défaut est 200 nœuds . Si un programme météo essaie d'installer une vitesse de vent supérieure (mais inférieure au WindDiscardLevel ci-dessus) , ça sera ignoré et 200 nœud sera installé à la place . Régler ce paramètre à 0 désengage complètement ce contrôle .

UpperWindGusts : « Yes » fait que FSUIPC4 retient toute information fournie par FS , ou par un programme météo externe , sur des rafales d'altitude . Elles sont d'ordinaire supprimées par FSUIPC4 parce que les vents d'altitude ne produisent pas de rafales . *Note : ce paramètre n'est pas opérationnel si* ***SuppressAllGusts*** *a été mis en service* .

SuppressAllGusts : réglez sur « Yes » si vous sentez que les rafales de vents fournies par FS ne sont pas réalistes, ou si tout simplement vous voulez quelques chose de plus simple ; Si c'est réglé sur « Yes » , alors le paramètre UpperWindGusts n'est pas activé .

WindTurbulence : « Yes » fait que FSUIPC4 génèrera des turbulence à tous les niveaux de vent . ça va de rien à extrême , mais normalement ça devrait rester gentiment au milieu ; ça variera en fonction du temps aussi bien .

SuppressWindTurbulence : « Yes » pour empêcher toute turbulence (mais pas les turbulences de nuages , il y a un paramètre dédié) .

SuppressWindVariance : « Yes » pour empêcher toute variation de vent (c'est-à-dire une instabilité directionnelle) ça affecte à la fois « Variable Winds » qui nous vient des réglages METAR, mais également

les variations de directions qui nous viennent à la fois du vent et des turbulences de nuages .

TurbulenceRate=1.0,5.0
————————————————
TurbulenceDivisor=20,20,40,40

Ça contrôle les imitations de l'effet vent de Wind smoothing . Le premier taux du numérateur est un multiplicateur pour la turbulence directionnelle , le second est le multiplicateur pour la vitesse de turbulence . la portée des deux est 0.0 à 10.0

Le premier nombre du dénominateur est le nombre d'étapes nécessaires pour changer la direction de la turbulence d'un extrême à l'autre (quelque chose à la survenue très improbable , mais ça commande la vitesse de tous les changements) , le second nombre est pour la vitesse de la turbulence , le 3^{e} nombre est pour la variabilité (variance) dans la direction du vent , et le 4^{e} pour les rafales de vent (de normales à max gust)

Le maximum dans les changements de direction du vent et sa vitesse dans des conditions de turbulences est obtenu en multipliant la valeur **TurbulenceRate** par le réglage de sévérité des turbulences dans FSX (0-4), et uniquement pour ce qui concerne la vitesse du vent , par 2% de la vitesse wind smoothed attendue (« normale ») . Ainsi , pour une vitesse de vent de 50 noeuds et des turbulences modérées (2) le paramètre Taux par défaut de 5.0 donne +/- 10 noeuds pour les changements de vitesse des vents .

Cette valeur représente l'extrême . Ensuite , FSUIPC4 calcule une cible au hasard en utilisant une approximation normale , Gaussienne , qui donne des valeurs de groupe (clustering) très proches de la normale .L'incrément calculé à partir du maximum et du paramètre Diviseur pertinent est utilisé pour déplacer la valeur actuelle (current) vers la nouvelle valeur . Lorsque ce but est atteint , une nouvelle cible est calculée , et ainsi de suite .

Notez que tout ceci est fait indépendamment pour ce qui concerne la direction du vent , de sa vitesse , de ses effets verticaux , de même que pour les rafales et la variance (qui , toutes deux ont une gamme de valeurs (range) imposée , bien sûr) . Les effets rafale et variance sont imités en utilisant des cibles avec une distribution normale , mais avec un écart-type (deviation) standard plus grand . permettant de la sorte

aux situations METAR extrêmes d'être atteintes , en temps réel , à l'occasion .

Le taux d'incrément est basé sur le taux de fluidité (frame rate) pour des turbulences , mais sur une moyenne de 5-10 Hz pour les rafales et la variance .

WindAjustAltitude : (toutes mes excuses pour la mauvaise orthographe) . A régler sur « Yes » au cas où FSUIPC4 ajouterait la valeur spécifiée dans **WindAjustAltitudeBy** à toutes les limites de couches de vents spécifiées par un programme météo externe utilisant l'interface FS98 (**pas** les interface Advanced ou New Weather) .

WindAjustAltitudeBy : voir le paramètre précédent . En pieds et par défaut jusqu'à FS2000 .

Windsmoothing : cette option commande le wind smoothing expérimental .

Windsmoothness : ce paramètre définit le degré ciblé de wind smoothing , avec une tendance à restreindre au maximum les changements de x noeuds et de x degrés par seconde . Le réglage par défaut est 5 (nœuds ou degrés) .

WindsmoothAirbornOnly : cette option de smooth seulement après décollage permet de changer les vents et de les régler comme on veut à l'aéroport , avant le décollage .

Visibilité

VisibilityOptions : détermine si les options Visibility sont mises en service ou non . Le réglage par défaut (« No ») désengage toutes ces options et cache également l'onglet pertinent dans la boite de dialogue on-line .

MinimumVisibility : le réglage par défaut est 0 (ce qui signifie : inactif) . Utilisé pour empêcher toute source météo d'installer une visibilité en-dessous d'un certain seuil . La valeur est calculée en centièmes de mile (c'est-à-dire 100 = 1 mile) Notez qu'il pourrait y avoir un délai court (autour d'une seconde) après qu'une nouvelle visibilité basse ait été appliquée avant qu'elle soit détectée et corrigée par FSUIPC4 .

MaximumVisibility : utilisé pour empêcher toute source météo d'installer une visibilité au-dessus d'un plafond maximum quand la couche de nuage est supérieure à 2/8^{e} . La valeur est définie en centième de mile (100 = 1 mile) . Notez qu'il pourrait y avoir un court délai après qu'une nouvelle visibilité ait été appliquée avant qu'elle ne soit détectée et corrigée par FSUIPC4 . Ce paramètre est efficace seulement si la valeur est supérieure à celle du paramètre MinimumVisibility .

MaximumVisibilityFewClouds : le même que le précédent , sauf que ce paramètre donne son maximum lorsque la couche nuageuse n'excède pas 2/8^{e} . L'idée est que la visibilité étendue donne un ciel plus bleu le jour et davantage d'étoiles la nuit (mais un taux de fluidité plus bas , désolé , vous ne pouvez pas gagner à tous les coups)

MaximumVisibilityOvercast : le même que le précédent , excepté qu'il donne son maximum quand la couche nuageuse est égale ou supérieure à 6/8^{e} .

MaximumVisibilityRaining : le même que le précédent , excepté qu'il donne son maximum quand il pleut ou qu'il neige. Si le temps est pluvieux et nuageux , alors les limites basses sont appliquées .

GraduatedVisibility : lorsque ce paramètre est mis en service , FSUIPC4 tente de fournir un changement doux dans la visibilité , depuis l'altitude la plus haute du niveau de visibilité de surface jusqu'à une visibilité spécifiée plus haute , à une autre , spécifiée, plus haute . Les deux paramètres , UpperVisibility et UpperVisAltitude commandent cela .

UpperVisibility : sur FS2000 et FS2002 ce paramètre , réglé par défaut à 6000 (équivalent à 60 miles) , est utilisé pour empêcher toute source. météo avec une visibilité réglée au-dessus d'un maximum spécifié . La valeur est réglée en centièmes de mile (100 = 1 mile) . Si GraduatedVisibility est mis en service , il est utilisé en conjonction avec le prochain paramètre (sur FS2004 également) .

LowerVisAltitude : utilisé seulement lorsque GraduatedVisibility est mis en service . Règle l'altitude à partir de laquelle la graduation devrait fonctionner . Normalement , vous laissez ce réglage à 0 , ce qui fait démarrer la graduation au sommet de la surface du plafond de visibilité

UpperVisAltitude : utilisé seulement lorsque GraduatedVisibility est mis en service Règle l'altitude à partir de laquelle UpperVisibility pourrait être atteint . Au-dessus de cette altitude , la visibilité reste fixée à cette valeur . Le réglage par défaut est : 25000 pieds .

ExtendMetarMaxVis : commande le réglage et les ajustements de la visibilité dans 3 circonstances :

1. s'il est réglé à une valeur comprise entre 99.95 et 100.04 miles , il est re-réglé à 6.20 miles , dans le but de rectifier les résultats de tous les programmes qui prennent la visibilité METAR de 9999 mètres et la transmettent littéralement comme un nombre exprimé en centièmes de mile .
2. si la valeur est alors comprise entre 6.15 et 6.24 miles (c'est-à-dire proche des 9999 mètres de la métrique METAR) elle est ajustée à une valeur au hasard comprise entre 6.20 miles et la valeur maximum actuelle (qui sera soit la valeur du paramètre MaximumVisibility , soit 150 miles .
3. si la valeur est comprise entre 9.95 et 10.05 miles (c'est-à-dire proche des 10 miles maximum de la métrique METAR US) alors elle est ajustée à une valeur au hasard comprise entre 10 miles et la valeur maximum actuelle (qui sera soit la valeur du paramètre MaximumVisibility , soit 150 miles) .

Notez que l'addition au hasard est calculée seulement toutes les 5 minutes , pour éviter que des changements constants de visibilité ne conduisent le programme météo à réécrire ponctuellement la valeur .

SetVisUpperAlt et **VisUpperAltLimit** imposent une limite supérieure au plafond de visibilité .

Nuages

GenerateCirrus : lorsqu'il est mis en service , crée occasionnellement une couche supplémentaire de cirrus .

OneCloudLayer : le réglage par défaut est « No » . Réglé sur « Yes » , il empêchera qu'il y ait plus d'une couche nuageuse , ce qui peut aider à obtenir de meilleures performances sur certaines machines plus lentes ; ça n'est pas d'une grande aide sur des machines plus rapides .

CloudTurbulence : Réglé sur « Yes » , fait en sorte que FSUIPC4 génère des turbulences au hasard dans toutes les couches nuageuses. La portée va de None à Extreme , mais normalement , ça se tient dans un juste milieu ; ça peut varier aussi bien en fonction du temps .

SuppressCloudTurbulence : Réglé sur « Yes » , empêche toute turbulence nuageuse à partir de toute source .

CloudIcing : Réglé sur « Yes » fait en sorte que FSUIPC4 génère une glaciation au hasard dans les nuages . La portée va de None à Extreme , mais normalement , ça se tient dans un juste milieu ; ça peut varier aussi bien en fonction du temps .

MaxIce : la valeur va de 0 à 4 (0 = pas de glaciation (icing) ; 4 = glaciation partout) avec un réglage par défaut à 3 (juste pour empêcher la glaciation « sévère » du niveau 4) . Si l'option est actuellement désengagée , la valeur est conservée par la sauvegarde d'un nombre négatif . Dans ce cas , -1 représente la valeur 0 , mais désengagée , jusqu'à -5 qui représente la valeur 4 , mais désengagée .

MinIce :la valeur va de 0 à 4 pour s'assurer que la glaciation n'est jamais en-dessous d'un niveau donné (<= 0 = Pas de glaciation ; 4 = Glaciation maximale) . Une valeur plus grande que MaxIce fonctionnera seulement pour mettre toutes les glaciations au niveau de MaxIce .

Autres options générales

PressureSmoothness : règle le nombre de centaines de hPa par seconde autorisés à changer . Lorsque ce paramètre est réglé sur 0 , l'option Pressure smoothing est désengagée .

TemperatureSmoothness : règle le nombre de centaines de degrés Celsius par seconde autorisés . Lorsque ce paramètre est réglé sur 0 , l'option smoothing de la température de l'air extérieur est désengagée .

AdvDisplayHotKey : vous permet d'assigner un appui touche de clavier qui , lorsque vous l'utilisez , cachera ou (s'il n'y a pas d'autre raison c'est caché) affichera la fenêtre message multilignes . L'appui touche est défini suivant les commandes de FS , listées ci-dessous , dans la section Programmation des Boutons . Par exemple j'utilise (et je

recommande) « CTRL+SHIFT+A » qui pourrait être : AdvDisplayHotKey= 65,11

AllEngHotKey : vous permet d'assigner un appui touche clavier qui , lorsqu'il est utilisé , re-sélectionnera tous les moteurs de l'avion actuellement chargé en mémoire . C'est effectivement la même chose que d'utiliser E suivi de 1 , 2 , 3 , ou 4 sur le clavier principal , suivant le nombre de moteurs , mais la hot key travaillera lorsque , apparemment , la séquence d'appui touches clavier ne fonctionnera pas (sur les avions à 3 moteurs me semble-t-il) . Voyez la précédente entrée pour des détails sur la façon dont la touche est définie .

DisconnTrimForAP : lorsque cette option est mise en service , FSUIPC4 déconnecte de FS l'axe du compensateur de gouverne de profondeur (elevator trim) selon que , soit : le pilote automatique FS est engagé en mode vertical (maintien de l'altitude ou pente de descente (glideslope) acquise) , soit : qu'un programme , une jauge ou un module a déconnecté l'axe du gouvernail de profondeur via FSUIPC4 (offset 310A) .

Notez que le réglage peut être transgressé (overridden) pour un avion particulier , avec des Calibrations de Joystick particulières avec des paramètres de réglage différents de ceux de la section [JoystickCalibration …] .

ZeroElevForAPAlt : commande à FSUIPC4 de centrer automatiquement la valeur imputée au gouvernail de profondeur chaque fois que le mode de maintien de l'altitude du pilote automatique est changé (on peut aussi changer en appuyant sur on ou off avec AP engagé)

Notez que le réglage peut être transgressé (overridden) pour un avion particulier , avec des Calibrations de Joystick particulières avec des paramètres de réglage différents de ceux de la section [JoystickCalibration …] .

MagicBattery : réduit le taux de décharge de la batterie en empêchant le voltage de se décharger (dropping) . Si c'est réglé sur « Yes » ou sur 0 , alors aucune décharge n'est tolérée . Si c'est réglé sur « No » ou sur 1 , alors la batterie se déchargera normalement . Toute valeur comprise entre 2 et 999 agira comme un diviseur du taux de décharge, ainsi 2 augmentera la durée de vie d'un facteur 2 , et ainsi de suite . C'est conçu pour aider à passer outre les erreurs apparentes dans les avions de

ligne , qui font qu'ils se déchargent beaucoup trop vite avant le démarrage des moteurs .

SetStdBaroKey : vous permet d'assigner un appui touche clavier qui , lorsqu'il est utilisé , règlera la fenêtre « Kollsman » sur l'Altimètre à la pression standard , 29.92'' ou 1013.2 mb . C'est utilisé en vol dans « Flight levels » .

L'appui touche est défini suivant les commandes de FS , listées ci-dessous , dans la section Programmation des Boutons . Par exemple j'utilise (et je recommande) « CTRL+SHIFT+B » qui pourrait être : SetStdBaroKey = 66,11

TCASid : FSUIPC fournit des données pour le TCAS externe ou les programmes de traçage à afficher , à l'avion AI supplémentaire qui vole dans le voisinage . Normalement , de tels avions sont identifiés par un numéro d'Avion de ligne et un numéro de Vol , s'il y en a un ; d'autre part par le numéro de Queue (tail number) .

Cependant d'autres types de chaînes (string) d'identification peuvent être choisies à la place . les labels optionnels placés sur l'avion par FS dans la scène affichent uniquement les numéros de Queue , ainsi si vous voulez les apparier , vous voudrez régler ces paramètres à « Tail ». L'Utilitaire « TrafficLook » peut montrer ces différences dans ses affichages . Voici la liste entière des options :

Flight pour Cie aérienne + Vol , ou numéro de Queue , disponible (par défaut)
Tail pour numéros de Queue seulement
Type pour « ATC Type » ,
Title titre de l'avion (dans le fichier .CFG) , tronqué à 17 caractères
Type+ pour le type au-dessus , tronqué soi nécessaire , plus les 3 derniers caractères du numéro de Queue
Model pour la description de Modèle .

TCASRange : règle la portée maximum à laquelle un avion AI peut être ajouté aux onglets (tables) pour les applications externes TCAS. Par défaut à 40 nm . Une valeur de 0 éteint complètement la limite .

FixedTCASOptions=Yes peut être ajouté par l'Utilisateur si les 2 réglages ci-dessus restent verrouillés (locked) , et inchangés excepté en éditant ici , dans le fichier .INI .

SetSimSpeedX1 : optionnellement , installe une HotKey qui , lorsqu'elle est utilisée , re-règle le taux de simulation à X1 (c'est-à-dire à la normale) . L'appui touche clavier est défini comme dans les commandes de FS, comme listée ci-dessous dans la section Programmation de Boutons . Par exemple , pour « CTRL+SHIFT+S » , ça pourrait être : SetSimSpeedX1=83,11

ThrottleSyncToggle : installe une fonctionnalité qui fait que toutes les entrées manettes des gaz , pour n'importe quel moteur , affectent les entrées manette des gaz de tous les moteurs . C'est une fonction interrupteur (toggle) – si elle est sur on , alors l'utiliser de nouveau la met sur off . Si vous n'utilisez qu'une seule manette des gaz , cela ne fera pas grande différence sinon qu'à chaque fois que vous utiliserez cet interrupteur , FSUIPC4 fera la sélection de manette des gaz (par exemple E+1 … etc .) appliqué à tous les moteurs. Par exemple pour « CTRL+SHIFT+E » , ça pourrait être ThrottleSyncToggle=69,11

ThrottleSyncAll : selon que le HotKey ThrottleSync fonctionne sur des valeurs Prop Pich & Mixture ou bien sur des manettes des gaz . ça n'a pas d'effet sur les Jets et les Hélicoptères .

FixControlAccel : lorsqu'il est mis en service , intercepte toutes les commandes et change la localisation du temps écoulé dans FS avant de faire suivre chaque commande *différente* (non-axe) , de sorte que le temps écoulé semble suffisamment grand pour que la commande ne soit pas accélérée . S'il voit des commandes identiques successives , alors il les quitte , ainsi elles peuvent être accélérées [**Ceci ne devrait pas être utilisé par ceux qui utilisent les commandes clavier**]

Pour une explication complète , reportez-vous au Guide de l'Utilisateur .

TimeForSelect : ça précise le nombre de secondes pour lesquelles la commande SELECT (normalement assignée au clavier principal , touches 1-4) devrait rester opérationnelle pour les commandes qui la nécessitent (comme Engine select ou Aircraft Exit toggle) , en dépit de l'intervention d'autres commandes , différentes . Pour désengager ceci , réglez le temps sur 0 Notez aussi que , alors que le réglage du Temps n'influence pas la fonctionnalité automatique similaire pour l'engin de manutention des avions (pushback) (qui fait en sorte que la direction du pushback reste sélectionnable) , cette fonctionnalité de FS sera également switchée sur off si 0 est précisé .

SpoilerIncrement : commande à partir de quel montant les commandes FSUIPC4 « Spoiler inc » et « Spoilerdec » changent la position de l'aérofrein à chaque utilisation . 512 par défaut , 32 crans de aérofreins non déployés (0) à pleinement déployés (16383)

AileronSpikeRemoval :
ElevatorSpikeRemoval :
RudderSpikeRemoval : ignorent tout signal venant de l'aileron , du gouvernail de d profondeur , et du palonnier , et qui spécifie le maximum de déviation (deflection) possible .

ClockSync : synchronise les valeurs des secondes avec celles de votre horloge système . Par défaut elle est sur « Off » (= No) Notez que la synchronisation peut fonctionner seulement lorsque les secondes = 0 , et alors on doit aussi régler les minutes . En conséquence on tentera seulement un ajustement lorsque la différence en minutes est moins grande que celle réglée par le paramètre suivant .

ClockSyncMins : la différence en minutes à l'intérieur de laquelle ClockSync fonctionnera . Le réglage par défaut est à 5 , mais si vous voulez réduire les rechargements en mémoire de Textures FSX , vous devrez le régler sur moins (less) . Au contraire , si vous voulez maintenir la valeur exacte en minutes aussi bien qu'en secondes , réglez-le à 59 ou 60 .

WhiteMessages : permet de faire suivre des applications externes à FS de sorte à ce qu'elles s'affichent en Blanc sur fond vert à la place de Rouge sur fond vert . Ceci s'applique seulement aux messages non déroulants (non-scrolling) [je ne suis pas sûr que ça fonctionne à présent]

UseProfiles : par défaut , réglé sur « No », pour des problèmes de compatibilité , mais réglez-le sur « Yes » si vous voulez utiliser la fonctionnalité User Profile plutôt que les assignements et calibrations spécifiques à un avion particulier . La fonctionnalité Profile a son propre chapitre dans le Guide de l'Utilisateur .

ShortAircraftNameOk : normalement réglé sur « No » pour être sûr que tous les réglages avion-spécifique ou profil-spécifique des Keys , Buttons et Calibration de Joystick s'appliquent seulement à l'avion spécifique qui a été chargé en mémoire pendant le temps qui lui a été assigné . Cependant , si vous avez plusieurs « peintures » et des réglages à appliquer à toutes , vous aurez besoin de régler ce paramètre sur

« Yes » puis raccourcir le nom de l'avion soit : dans la section [Profiles] si vous utilisez des Profils , ou bien dans la section de [Axes.<name>] , [Buttons.<name>] , [Keys.<name>] et [JoystickCalibration.<name>] dans le fichier .INI , en fonction des besoins . La même fonctionnalité pourrait , par exemple , donner à tous les avions commençant par « Boeing » un lot d'assignement et à tous ceux commençant par « Airbus » un autre lot .
De plus , vous pouvez régler **ShortAircraftNameOk=Substring** pour faire en sorte que FSUIPC4 assortisse le <nom> raccourci dans les profils INI ou les en-tête de section à n'importe quel endroit de la totalité du nom de l'avion , pas seulement au début du nom .

ShowPMcontrols : ça rappelle en mémoire simplement les réglages Magenta Project pour les assignements déroulants (drop-downs) .

ReversedElevatorTrim : probablement pas d'un usage effectif aujourd'hui , tous les axes peuvent être inversés dans les fonctionnalités de calibration de Joystick de FSUIPC4 . Il vaut mieux le laisser sur « No » .

Notez que vous pouvez passer outre ce réglage pour un avion particulier avec des calibrations particulières de Joystick en réglant ce paramètre différemment dans la section [JoystickCalibration …]

PauseAfterCrash : c'est l'option Miscelleanous pour régler FSX sur Pause après qu'il ait rechargé en mémoire un vol suite à un crash . ça permet à l'avion d'être déplacé sans danger , en le sortant d'une boucle infernale .

LoadFlightMenu et **LoadPlanMenu** enregistrent simplement le choix sur la page d'option Miscelleanous , pour les entrées menu Add-Ons , pour charger en mémoire respectivement les vols et les plans .

ZapSound : définit le son devant être utilisé lorsque la commande FSUIPC pour la suppression du trafic AI (le « Traffic Zapper ») est appliquée avec succès . ça doit être le nom d'un fichier .WAV dans le répertoire son de FS , le fichier par défaut étant « Firework » .

Si vous ne voulez pas un son , alors réglez juste sur **Zapsound=None** . Cependant , la raison pour le son est que de cette façon , vous savez que quelque chose a été zappé FSUIPC ne peut pas vous dire ce que vous pouvez voir , et l'avion qui est zappé peut ne pas être à l'écran , de telle sorte que vous ne pouvez pas le voir disparaître …

ZApAirRange=1.5
ZapGroundRange=0.25

Commande la portée de zappage d'avion AI . Les unités sont en miles nautiques . Air et Ground se réfère à la localisation de l'Utilisateur , pas à la cible . Notez que vous ne pouvez pas changer l'angle de réception (acceptance) de façon explicite . C'est ajusté automatiquement , en proportion linéaire inverse au changement dans la portée- ainsi avec une portée plus grande , vous aurez besoin de pointer le nez de l'avion de façon plus précise .

ZapCylinderAltDiff=n (où n est la différence maximale d'altitude) , peut être ajouté pour changer le mode du Zappeur en vol (airborne)Une fois ceci ajouté , la cible à zapper est l'avion le plus proche de l'Utilisateur en vol , qui est dans la zone droite d'un cylindre de rayon ZapAirRange , et qui a une différence en altitude de n pieds ou moins , avions au sol inclus .

MouseWheelTrim : enregistre le réglage de l'option » Use mousewheel as trim » de l'onglet Miscelleanous . réglé par défaut sur « No » .

SaveDataWithFlights=Never : enregistre l'onglet Miscelleanous , disant à FSUIPC s'il doit sauvegarder les données « offset » lorsque les vols sont sauvegardés , ou pas , et quand les recharger en mémoire . Les options sont **Never , Menu , Auto** et **Yes** (« Yes » étant l'option affichée comme « Always » dans l'onglet) .Les fichiers .IPCBIN sont toujours sauvegardés avec les vols dans tous les cas excepté le mode **Never** .

Pour une explication complète des différences entre les options , voyez , je vous prie , le Guide de l'Utilisateur . Pour les techniciens et les programmeurs , notez que les fichiers .PICBIN produits contiennent une photo de la gamme entière (65536) des « offsets » FSUIPC , ce qui fait que toutes les sortes de données peuvent être lues à partir d'eux en utilisant un éditeur hexadécimal .

BrakeReleaseThreshold=75 :s'utilise lorsque votre freinage est commandé par des pédales plutôt que par des touches clavier ou des boutons de joystick assignés à des commandes de freins non-axe (non-axis) . Dans ce dernier cas , faire fonctionner les freins automatiquement desserre le frein de parking (et il est possible qu'il efface également l'action d'autofreinage) ça ne doit pas arriver normalement avec des axes de freins utilisés pour freiner , puisque ce sont des commandes séparées . ça pourrait être vu comme un

inconvénient d'avoir un mode de freinage avec les pieds , c'est pourquoi ce paramètre est fourni pour régler la quantité de freinage nécessaire pour desserrer le frein de parking . Le nombre est un pourcentage du freinage total – donc par défaut , 75% . Si vous le réglez à 0 % , il coupe la fonctionnalité . Pressez sur les deux freins avec au moins le niveau de réglage , l'action de desserrer ne sera pas réarmée tant que les deux freins retourneront sur « off » . Les freins à pied doivent être calibrés tous les deux dans FSUIPC4 .

JoystickTimeOut=20 : ce temps mort n'est plus applicable , excepté pour les périphériques EPIC USB , et vous pouvez désormais les ignorer .

BlankDisplay=No : A régler sur « Yes » si vous voulez FSUIPC4 à vide (to blank) et des affichages GoFlights connectés Durant l'initialisation de FS .

PollGFTQ6=Yes : à changer pour « No » pour stopper le sondage du module GoFlight TQ6 par FSUIPC4 . Il semblerait que tous les axes et boutons de ce périphérique soient toujours manipulés depuis Windows en tant que périphérique joystick , donc avoir FSUIPC ainsi scanné donne des indications doubles dans l'onglet Buttons de FSUIPC4 (vu comme Joystick 169) .

Options techniques moins utilisées

AutoTuneADF : pour « auto-tuner » la radio ADF . Si cette commande est mise en service , alors FSUIPC détecte que s'il ne reçoit aucun signal NDB , alors il emploie tour à tour la partie fractionnée de la fréquence ADF entre .0 et .5 toutes les 7 secondes ; ça permet aux tableaux de bords (cockpits) externes construits avec des fonctionnalités radio ADF en nombres entiers d'être utilisés dans des endroits comme le Royaume Uni qui a beaucoup de fréquences NDB qui se terminent par .5 .

AxisCalibration : cette fonctionnalité traite des entrées vers les offsets des axes de palonnier , d'aileron et d'élévateur , via les offsets IPC . Conçu pour être utilisé avec les pilotes hardware qui , au lieu d'envoyer des entrées axes normales à FS , commande les surfaces du vol principal par des écritures directes vers les offsets de FSUIPC, donc en court-circuitant les assignements , calibrations etc …Le seul pilote

hardware que je connaisse est celui pour la console GA28R d'Aérosoft . Dans FSUIPC4 , cette fonctionnalité fonctionne encore par compatibilité avecFSUIPC3 et versions en-dessous . **Cependant , vous êtes avertis de la régler sur « No » , et d'utiliser à la place la nouvelle fonctionnalité « DirectAxesToCalibs »**

DirectAxesToCalibs : réglé sur « Yes » , fait que FSUIPC4 suppose que toutes les écritures directes vers des offsets de FSUIPC4 pour le palonnier , l'aileron et l'élévateur viennent d'un pilote hardware et sont en réalité comprises comme des entrées axes . FSUIPC4 dirige les valeurs vers sa section Joystick Calibrations , où vous pourriez alors calibrer les entrées comme vous le désireriez . Le seul hardware que j connaisse qui bénéficie de cette structures et la console GA28R d'Aérosoft . Ne réglez pas cette option si vous utilisez tout tableau de bord sophistiqué ou programmes externes avec leurs propres pilotes automatiques (autopilots) puisqu'il est possible qu'ils acheminent leurs valeurs de commandes de la même façon . FSUIPC4 ne peut pas distinguer la source .

AxisIntercepts : peut être réglé sur « Yes » pour forcer l'acheminement et l'interception des commandes axes par FSUIPC4 , même si cette action n'est pas nécessaire pour la calibration de FSUIPC4 (où ce sera fait automatiquement dans tous les cas) . Cette action sera nécessitée seulement par quelques avions « fly-by-wire » .

StartImmediatly : on n'attend pas qu'elle soit utilisée directement par l'Utilisateur . Lorsqu'elle est réglée sur « Yes » , elle fait en sorte que FSUIPC4 initialise l'interface données avec Simconnect dès son démarrage , plutôt que d'attendre que SimConnect indique « SimStart » . C'est le réglage par défaut pour FSX + SP1 ou versions postérieures . De sorte à éviter les problèmes Simconnect plus anciens qui surviennent lorsque plus d'un Client initialise en même temps , le réglage par défaut se fait sur « No » pour les versions antérieures à FSX + SP1 .

SimConnectStallTime : peut être utilisé sur des systèmes à faibles performances , très lourdement chargés en mémoire , pour essayer d'empêcher FSUIPC4 de réinitialiser l'interface SimConnect à FSX quand il se trouve affamé d'informations qui devraient arriver tout le temps . Le réglage par défaut à une seconde (1) , qui pourrait être toujours adéquat pour les systèmes qui peuvent maintenir une cadence (frame rate) de 2 fps ou plus , ne tombe jamais à 1 ou en-dessous . ça n'inclut pas les périodes où FSX recharge une scène en mémoire , ou dans des Menus , etc .. , qui ne sont pas vérifiés (checked) .

La gamme des valeurs va de 1 à 9 , 1 étant le réglage par défaut .

UseEpsilon : est normalement complètement omis . Si nécessaire , il peut être ajouté dans la section [General] du fichier .INI , réglé sur « Yes » . ça imposera de changer des limitations sur beaucoup de variables envoyées par SimConnect à FSUIPC4 . Ces restrictions étaient en place dans FSUIPC4 avant la version 4.069 , mais elles furent enlevées par défaut quand on détermina que le gain en performance était nul ou inexistant .

FiddleMachForPM=Yes est heureusement un correctif (fix) temporaire pour les Utilisateurs du Projet Magenta qui souffrent d'une décroissance de hautes vitesses sous commande MCP , résultat d'une vitesse Mach réglée de façon non correcte dans le mauvais mode . ça affecte seulement FSX à cause de la façon étrange dont fonctionnent FS9 et les versions antérieures . Pour mettre en service ce correctif , ajoutez ce paramètre (il ne devrait pas apparaître dans le fichier INI par défaut) .

En manière d'explication , quand le MCP PM règle l'IAS pour le A/P , il règle également la valeur Mach . ça pourrait être OK , excepté pour 3 choses :

1- la valeur Mach qu'il règle après Cruise est TOUJOURS la dernière qu'il règle dans cruise , pas celle , correcte , qui se réfère à l'IAS et à altitude/température/pression
2- le Mach est écrit après l'IAS
3- dans FS9 et versions antérieures , le réglage Mach n'affecte pas le réglage IAS et vice et vers , jusqu'à temps que , et tant que vous switchez les modes IAS/Mach .Cependant , dans FSX , la dernière valeur écrite s'applique et est convertie dans son équivalent Mach/IAS peu importe le mode dans lequel vous êtes !

Ce paramètre FSUIPC4.INI fait en sorte que FSUIPC4 abandonne tout simplement toute écriture MACH pendant que le MCP PM est en mode IAS .

FiddleAppAltForPM=Yes :

Un autre correctif temporaire heureux pour le Projet Magenta . Il fait en sorte que FSUIPC4 remplace automatiquement toute altitude écrite durant le modes APP du MCP de PM par 0 . Il règle aussi l'altitude MCP de FSX à 0 dans le mode APP du MCP de PM . lorsqu'une VS négative

est réglée , et il fait les deux choses en même temps ,même si le maintien de l'altitude est mis en service .

L'intention est ici d'évier des remontées intempestives lors d'un alignement de descente (Glideslope) , dû au fait que FSX semble être différent de FS9 et versions antérieures , en ce que écrire dans le registre l'altitude du MCP de FS peut affecter la vitesse verticale requise , même si l'option « maintien de l'altitude » de FS n'est pas mise en service .

AutoGlobalWeather=Yes contrôle si FSUIPC4 tente de changer les structures globales météo internes à FSX lorsque les options météos variées (assorted) sont appliquées . Généralement , ça ne semble pas trop causer de tort lorsque c'est laissé en service , mais le bien que ça peut faire n'est pas vraiment établi .

CustomWeatherModify=No stoppe la réécriture par FSUIPC4 de la station météo de METAR dans une tentative de corriger des vents et des températures excessifs et aussi d'appliquer des filtres météo sélectionnés lorsque le mode « custom weather » est mis en service – comme lorsque l'Utilisateur sélectionne des réglages météo particuliers (c'est-à-dire non thématiques ou téléchargés) ou lorsque des programmes externes comme ASX contrôlent la météo .

AutoDeleteAI : ça contrôle une fonctionnalité qui fait en sorte que FSUIPC4 détruit automatiquement un avion AI avec des signes d'appel donnés (ID de ATC) . Le paramètre fournit une liste (jusqu'à 16) de signaux d'appel (call signs) , mis entre guillemets de façon facultative et séparés par des virgules .Comme ce sont les remorqueurs de planeurs (glider tow planes) qui sont a raison pour laquelle a été ajoutée cette fonctionnalité, et que leurs signaux d'appel semblent toujours être EC-527 , la ligne pourrait être :

AutoDeleteAI=EC-527

Notez que pour les avions remorqueurs (tow planes) , ça détruit l'avion AI , mais pas le filin de remorquage. Apparemment , vous auriez besoin de le déplacer (Ctrl Y ?) de sorte à ce qu'un autre avion remorqueur soit appelé .

L'action de détruire ces avions est « Off » , initialement , par défaut , mais elle peut être réglée sur « On » par défaut en mettant « ON » comme premier mot de la liste , donc :

AutoDeleteAI=ON,EC-527

Ça peut être utilisé sur un serveur multi-joueurs , où d'autres avions remorqueurs ne sont probablement jamais appelés. La commande de la durée d'exécution de l'état de cette option se fait par l'utilisation de 3 nouvelles commandes , ajouté au menu déroulant de FSUIPC4 pour Buttons & Keys .

UseAxisControlsForNRZ=No : une fonctionnalité pour la section [JoystickCalibration] du fichier.INI , pas pour la section [General] . C'est une option spéciale pour essayer de s'adapter à différentes pratiques d'add-ons (notamment , dans ce cas , le Wilco A320) . Normalement , les calibrations des 4 manettes des gaz , 4 Mixtures et 4-Prop pitch consistent en une sortie (output) avec , soit : une gamme qui inclut la zone reverse , ou bien , si l'option « no reverse zone » est cochée , une portée de 0 (idle) à 16383 (max) . Sorties qui sont envoyées à FS en utilisant les vieilles commandes « ????n_SET » , (THROTTLE1_SET , etc ..) puisque ce sont celles qui fournissent la zone de poussée inversée (reverse zone) en-dessous de zéro .

Si vous réglez le paramètre .INI **UseAxisControlsForNRZ** de [JoystickCalibration] sur « Yes » , alors l'option NRZ (no reverse zone) pour les 3 types d'axes utilisera à la place les commandes AXIS_ ????n_SET (par exemple AXIS_THROTTLE1_SET) , avec une portée de – 16383 (idle) à + 16383 (Max) . ça sera avion-spécifique ou profil-spécifique selon que vous le règlerez dans la section calibration appropriée du fichier .INI .

AUTOSAVE : uniquement fichiers INI

Quelques programmes add-on produisent des fichiers lorsque les Vols sont sauvegardés séparément des vols habituels dans le répertoire Vols de FSX , de telle sorte que l'option AutoSave échoue à gérer leurs nombres , détruisant les plus vieux d'entre eux lorsque les fichiers .FLT sont détruits . Les types maintenus (handled) par défaut sont : FLT , WX , FSSAVE , PSS , FMC , ABL , RCD , SPB et IPCBIN . Pour quelques autres , et des fichiers dans d'autres répertoires , vous devez ajouter manuellement quelques lignes à la section [AutoSave] du fichier FSUIPC4.INI . Par exemple , pour le PMDG 747 FSX , ça pourrait être :

AlsoManage1=PMDG\747400\PanelState*.FLT.sav
AlsoManage2=PMDG\747400\PanelState*.0.rte
AlsoManage3=PMDG\747400\PanelState*.1.rte

Le chemin donné dans ces lignes et à l'intérieur du chemin principal FSX . Si vous avez quelque chose d'installé en dehors du chemin FSX , vous aurez besoin de donner le nom de chemin complet , à partir du lecteur (par exemple C:\...) (ou du nom de l'ordinateur pour un Réseau dans la forme habituelle , c'est-à-dire \\<name> …)

Fonctionnalités de Logging

Ces options peuvent être commandées « en vol » à partir de la fenêtre de la boite de dialogue de FSUIPC4 (sélectionnez le menu Add-ons , puis FSUIPC , ou ALT , D , puis F) .

FSUIPC4 produit toujours un fichier texte appelé FSUIPC4.LOG dans le répertoire Modules . Les entrées dans le Log sont chronométrées (timed) à partir du démarrage de la session FS. Les temps , en millisecondes , apparaît à l'extrême gauche de chaque ligne .

Utilisez s'il vous plaît les fonctionnalités de logging pour vérifier des choses avant de signaler les problèmes ou les omissions fans FSUIPC4 et de fournir (ou d'extraire) un fichier log approprié proprement dé-zippé qui contient es signalements .
Notez que les fichiers log peuvent devenir très grands si toutes les options sont en service . Gardez les tests de vols brefs . Vous pouvez lire les fichiers log puisque voler vous permet d'utiliser un lecteur qui partage l'accès (comme les programmes récents Notepad) , ou qui utilise la « NewLogKey » décrite ci-dessous pour fermer des logs et en ouvrir de nouveaux .

Tous les paramètres de commandes Log vont dans la section [General] du fichier FSUIPC4.INI . Aucun n'est inclus par défaut .

LogWeather=Yes : données Log météo envoyées par un programme de commandes météo , et les données actuelles météo construites par FSUIPC4 en termes FS . Alors vous obtenez la météo lue par FSUIPC4 et placée en dernier lieu en arrière dans les Offsets pour des applications à lire . Les données de commande de la météo à venir sur

les interfaces Afvanced Weather et New Weather sont également enregistrées (logged) à plein .

LogWrites=Yes : enregistre (log) les « écritures » offset reçues à partir des applications , avec des adresse globales offset et des tailles de données , plus tous les bytes des données . Les offsets montrés sont ceux qui sont utilisés par les applications [attention , le fichier log peut devenir très grand !]

LogReads=Yes : enregistre (log) les « écritures » offset reçues à partir des applications , avec des adresse globales offset et des tailles de données , plus tous les bytes des données . Les offsets montrés sont ceux qui sont utilisés par les applications [attention , le fichier log peut devenir très grand !]

LogEvents=Yes : cette option enregistre tous les « événements touches » (key events) , autres que ceux à partir de commandes d'axes . ça peut être très utile à ceux qui cherchent à comprendre les actions de leurs boutons et touches clavier (keys) , ou à voir les sortes de choses que font les tableaux de bord les plus complexes ; de façon répétée , chaque seconde .

LogAxes=Yes : ça enregistre juste les événements d'entrée d'axes .

LogButtonsKeys=Yes : ça enregistre la plupart des événements clavier (les KEYUPs seulement lorsqu'ils sont programmés) , et toutes les opérations Boutons . L'enregistrement peut être assez long , mais il sera très utile lorsque vous essaierez d'analyser exactement ce que font votre Bouton complexe FSUIPC4 ou votre touche programmée .

LogLua=Yes : met en service l'enregistrement de modules d'extension supplémentaires Lua , et incite chaque module d'extension Lua en service (running) à utiliser ses propres fichiers Log , qui sont cumulatifs – quoique chaque fois que le même module d'extension est en service il s'ajoute au fichier Log existant .

LogExtras=Yes : enregistre des données techniques additionnelles au sujet des travaux internes à FSUIPC4 dont la nature variera de temps en temps en accord avec les besoins . Il n'y a rien d'intéressant pour l'Utilisateur , mais leurs problèmes d'investigation peuvent les amener à le mettre en service , de sorte que les logs retournés deviennent plus explicites quant à leur résolution . Ne volez pas de façon extensive avec

cette option en service ou vous ferez le plein de votre disque et vous compromettrez probablement sa performance.

Des fonctionnalités d'enregistrement supplémentaires « extra » sont disponibles si le paramètre **Debug=Please** est incorporé au fichier .INI . ça change les drapeaux (flags) enregistrés extras en une valeur numérique qui peut être en hexadécimal et peut s'étendre sur deux parties de 32 bits (x….,x ….) . Cette fonctionnalité s'utilise uniquement sous instruction .

LogSimC=xxxx,xxxx (où chaque « xxxx » est soit un offset , ou une gamme xxxx-xxxx) selon que les valeurs associées avec les offsets listés ou inclus ici sont lues à partir ou écrites à une variable SimConnect (« SimVar »), ces valeurs sont enregistrées . La liste peut demander plusieurs offsets disparates ou gammes – la limite est imposée seulement par la longueur maximum des lignes du fichier .INI (255 caractères) .

NewLogKey, StopLogKey : vous permet d'assigner des appuis touches clavier pour fermer le fichier Log actuel (si logging a été mis en service) , et en démarrer un nouveau . Le « NewLogKey » poursuivra les mêmes options logging , alors que « StopLogKey » reviendra au logging par défaut (le minimum) Ces deux clés donnent à elles deux un contrôle complet du logging . (Notez que les deux actions sont également disponibles dans la fenêtre dialogue de FSUIPC4) .

Le fichier log actuel est toujours appelé FSUIPC4.LOG Les autres sont nommés par ordre numérique , FSUIPC4.1.LOG , …2.LOG, … etc . Les appuis touches sont définis comme dans les commandes de FS et listés ci-dessous dans la section Programmation des Boutons . par exemple , j'utilise « Shft+Ctrl+L » et « Shft+Ctrl+O » (pour « log » et « off » respectivement) ce qui pourrait être :

NewLogKey=76,11
StopLogKey=79,11

Fonctionnalités monitoring

FSUIPC4 peut superviser (monitor) , sur chaque trame (frame) de FS , jusqu'à 4 valeurs (ou les mêmes valeurs dans différents formats , si besoin) , et les afficher ou les enregistrer lorsqu'ils changent . Pour

chaque valeur à enregistrer , vous devez saisir ou sélectionner 4 choses :

Base : qui sera normalement fixée à « IPC » . La base est le nom de l'aire de données à partir de laquelle la valeur montrée sera prise . Toutes les variables supportées par FSUIPC à travers l'interface IPC sont à offsets en rapport avec la base IPC .
Offset : qui identifie la position de la valeur relative à la Base . C'est un nombre hexadécimal , qui va normalement de 0000 à FFFF . Quelques unes des Bases non-IPC peuvent permettre des offsets plus grands . Pour les offsets aux variables standard IPC , voir le Guide des Programmeurs dans le SDK .

Type : ça définit le type de variable , de telle sorte que le formatage dans l'affichage montrera quelque chose de pleinement significatif . Les types actuellement supportés sont dans le tableau ci-dessous :

Type	**Description**	**C Type**
S8	Valeur signée 8 bits , -128 à +127	**signed char** (caractère signé)
U8	Valeur non signée 8 bits , 0 à 255	**unsigned char** ou **BYTE**
S16	Valeur signée 16 bits (2 byte)	**short**
U16	Valeur non signée 16 bits (2-byte)	**unsigned short** ou **WORD**
S32	Valeur signée 32 bits (4-byte)	**int**
U32	Valeur non signée 32 bits (4-byte)	**unsigned int** , ou **DWORD**
SIF16	2 byte Entier&Fraction : fraction 8 bit suivie par Entier 8 bit signé	Utilise un **short**
UIF16	2 byte Entier&Fraction : fraction 8 bit suivie par Entier 8 bit non signé	Utilise un : **unsigned short**
SIF32	Entier&Fraction 4byte : Fraction 16-bit et Entier signé 16-bit	Utilise un **int**
UIF32	Entier&Fraction 4byte : Fraction 16-bit et	Utilise un **unsigned int**

	Entier non signé 16-bit	
SIF64	Entier&Fraction 8byte : Fraction 32-bit et Entier signé 32-bit	Utilise un **: unsigned** puis un signed **int**
UIF64	Entier&Fraction 8byte : Fraction 32-bit et Entier non signé 32-bit	Utilise deux unsigned ints
FLT32	Valeur de point flottant standard 32-bits (4-byte)	**Float**
FLT64	Valeur de point flottant standard 64-bits (8-byte)	**Double**
ASCHZ	Une chaîne de caractères single-byte terminé par zéro byte. Une longueur d'un nombre limité est montrée	**Char[ ]** ou **ASCHZ**
SA16	Angle signé -16bits en format FS (-180 degrés = max+1)	Utilise un **short**
UA16	Angle non signé -16bits en format FS (360 degrés = max+1)	Utilise un **unsigned short**
SA32	Angle signé -32bits en format FS	Utilise **int**
UA32	Angle non signé -32bits en format FS	Utilise **unsigned int**

Hex : pour la plupart des valeurs numériques , l'affichage sera décimal . Cependant , pour des valeurs d'entiers pleins à point fixe (S8 , U8 , S16 , U16 , S32 et U32) vous pouvez les voir en hexadécimal si vous le souhaitez .

Vous devrez alors sélectionner comment vous voulez que les valeurs s'affichent . Il y a 4 options , qui peuvent toutes être sélectionnées :

Normal Log File : des changements dans les valeurs monitorées sont listées dans le fichier FSUIPC4.LOG pour être vues plus tard . De plus , pour chaque offset monitoré , l'offset est traité également

automatiquement en tant que offset « LogSimC » (voir ci-dessus) de telle sorte que les lectures/écritures SimConnect sont enregistrées .

Debug String : les mêmes messages sont envoyés à un débuggeur comme DebugView , pour voir en parallèle aux actions FS . Notez que vous pourrez avoir des difficultés à lancer un débuggeur avec les versions safe-disk protégées de FS (FS2000 et FS2004) .

FS Window : le monitoring est effectué en utilisant jusqu'à 4 lignes dans la fenêtre d'affichage des messages FS .

FS Title Bar : les messages remplacent complètement la barre de titre de FS . Un seul est affiché à la fois , donc c'est utile seulement pour monitorer une valeur .

Si la valeur requise n'est pas disponible à chaque instant , le résultat affichera <invalid> Lorsque vous recherchez des moteurs ou d'autres choses qui concernent un avion , cela peut arriver de façon éphémère , par exemple pendant qu'un avion est chargé en mémoire ..

Toutes les sélections de monitoring sont sauvegardées dans le fichier FSUIPC4.INI , dans une section appelée [Monitor] .

JoyNames

La section [JoyNames] du fichier INI est décrite de façon exhaustive dans son propre chapitre , dans le Guide de l'Utilisateur .

Profiles

Si vous optez pour l'utilisation de Profils , pour avoir des réglages de différents Boutons , Touches clavier , Axes et Calibrations , dédiés à un nombre de type d'avions (plutôt que des avions particuliers) , alors FSUIPC4 créera des sections [Profile.<name>] dans votre fichier .INI . Ils prennent le nom du Profil que vous demandez , par exemple : « Jets » , « Props » , « Hélicos » , et contiennent simplement une liste , de format habituel 1=<name> , 2=<name>… des avions dont le nom appartient à ce profil particulier , en accord avec vos assignements . Ces noms

d’avion peuvent être des noms entiers , comme lorsque vous les assignez dans la boite de dialogue de FSUIPC4 ; ou bien ils peuvent être raccourcis ou en sous-chaînes (substring) , en accord avec le paramètre « ShortAircraft\NameOK » déjà mentionné .

Programmation Boutons

FSUIPC4 fournit une page pour la programmation Boutons dans le Menu principal . Nous examinons ici comment ce programme est encodé dans le fichier FSUIPC4.INI et comment le programme peut être étendu pour fournir de multiples appui touches clavier et commandes pour un Bouton , mélangés au besoin , et pour fournir des actions composées (conditionnelles) – qui dépendent d’autres boutons , de réglages d’interrupteurs et même d’appuis touches clavier précédents . Il existe même des fonctionnalités pour faire dépendre les actions de Boutons de valeurs en offsets à partir de l’interface IPC de FSUIPC4 , qui fournit réellement une pléthore de possibilités (pour cette partie vous aurez besoin d’acquérir le FSUIPC4 SDK , puisque la liste des offsets est fournie dans ce package , dans le Guide du Programmeur) .

FSUIPC4 recharge en mémoire tous les paramètres Boutons chaque fois que l’avion est changé dans FS, donc vous pouvez les éditer et les tester sans avoir besoin de recharger FS en mémoire à chaque fois .

Avant de s’embarquer sur la programmation elle-même , plusieurs paramètres globaux ont besoin d’être décrits . Ceux-ci n’apparaîtront pas dans le fichier .INI tant que vous ne les aurez pas ajoutés , et vous aurez besoin de les y ajouter (dans la section principale [Buttons]) que si vous avez besoin de quelque chose d’autre que les réglages par défaut :

InitialButton : fait en sorte que FSUIPC4 initie des actions on-off lorsque FS est chargé en mémoire pour la première fois ou bien lorsqu’il est lancé pour la première fois (c’est-à-dire prêt à voler) . ça se fait en programmant un Bouton réel ou imaginaire . Ajoutez simplement la ligne : « InitialButton=j,b »à la section [Buttons] . Les valeurs de j (0-255) et de b (0-31) peuvent spécifier un Joystick ou un Bouton , réels ou imaginaires , ça n’a pas d’importance . Les réels peuvent avoir une action assignée on-line , dans la page Buttons , mais des actions multiples pour chaque Bouton , réel ou non , peuvent être accomplies en éditant le fichier .INI comme décrit ici .

IgnoreThese : peut être utilisée pour lister un nombre de boutons qui doivent être ignorés par FSUIPC4 dans l'onglet Buttons & Switches . C'est fait pour s'occuper des signaux de Boutons défectueux qui se répètent sans commande et empêchent les autres d'être inscrits sur l'écran prêt à programmer. Le paramètre prend cette forme :

IgnoreThese=j.b,j.b,…

Listant le nombre joystick (j) et le nombre Bouton (b) de chaque Bouton devant être ignoré . Pour rendre ceci facile , vous pouvez éditer le fichier .INI pendant que vous êtes dans la boite de dialogue des assignements de Boutons et simplement presser sur « reload all buttons » pour activer les changements .

Notez que l'action d'ignorer les Boutons s'applique seulement à ceux numérotés de 0 à 31 sur chaque joystick possible (pas sur des chapeaux de « POV ») et qu'ils sont seulement ignorés dans la boîte de dialogue –s'ils sont toujours assignés , leur assignement sera encore effectif .

EliminateTransients : ça peut être ajouté , et réglé sur « Yes » pour éliminer des indications courtes (éphémères) d'appui boutons . C'est destiné à fournir de l'aide pour le traitement de certains périphériques qui créent des signaux d'appui bouton sans fondements . ça fonctionne seulement avec des joystick connectés localement (mais pas avec les périphériques EPIC ou GoFlight) .

Notez que mettre en service cette option peut signifier que vous aurez à appuyer consciencieusement sur les boutons pendant un intervalle de temps légèrement plus long . ça dépend du **PollInterval** (ci-dessous) . Une indication de bouton « transient » n'existe que pour un sondage (poll) , donc une pression réelle devrait durer au moins 50 mSec (deux fois le poll interval par défaut) pour être sûr d'être perçue (plus pour autoriser des variations dans le sondage dues à l'activité de FS versus le processeur . Vous pourrez penser qu'il vous faut ajuster le **PollInterval** .

PollEpicButtons=Yes : à régler sur « No » si vous faites l'expérience de quelque difficulté à obtenir que FSUIPC4 fonctionne correctement sur un système avec un périphérique EPIC installé , que vous ne voulez pas programmer via la page « Buttons » de FSUIPC4 .

ButtonRepeat=20,10 : le premier nombre commande le taux de répétition du Bouton , lorsque répéter est mis en service pour un Bouton

spécifique . Le nombre de répétitions par seconde va de 1 à 100 . Notez que les taux les plus hauts ne peuvent pas , actuellement , être réalisés . Si vous voulez permettre que les répétitions se produisent aussi vite que possible quelles que soient les circonstances , réglez ce paramètre à 0 . Elles peuvent être très rapides , alors attention ! …
Notez qu'il est peu probable que ce taux soit exactement maintenu puisqu'il dépend des variations de performance de FS en fonction de l'action à répéter , mais il agit comme une bonne valeur cible de commande .

Le second nombre donne un délai initial , avant le début des répétitions . C'est du point de vue de combien de répétitions potentielles ce délai va représenter , donc avec 20 par seconde , 10 donneraient un délai d'une demi-seconde . ça permet au même bouton de fonctionner pour augmenter/diminuer une valeur juste une fois , ou , en maintenant le bouton appuyé , à répéter jusqu'à ce que le bouton soit relâché .

Une valeur de 0 pour le délai initial signifie qu'il n'y aura pas de délai avant que la répétition ne commence – c'est-à-dire l'état de FSUIPC avant que le délai ne soit ajouté .

PointInterval=25 : ce paramètre dit à FSUIPC comment lire (« sonder ») les boutons de joystick . le temps est en millisecondes , et la valeur par défaut est 25 (40 fois par seconde) .

Un taux de sondage de 0 stoppera la recherche complète de Boutons de FSUIPC4 , et en fait ça déplacera l'onglet Buttons & Switches de FSUIPC4 . ça peut se révéler utile pour contrôler si un pilote de joystick coquin ne cause pas de problèmes .

Un taux de sondage de 40 par seconde est plus qu'adéquat pour toutes les programmations de boutons normaux . C'est seulement lorsque vous en venez à des utilisations plus sophistiquées que vous voudrez changer cela . Les interrupteurs à boutons rotatifs , par exemple , peuvent donner des pulsations si rapides que quelques unes peuvent se perdre en route .

Toute valeur à partir d'une milliseconde peut être spécifiée , mais toute valeur à partir de 50 entraîne un nombre spécifique de « ticks » (55 mSecs) , par exemple 40-82 impliquent actuellement 55 (1 tick) , 83-138 , 2 Ticks , etc … Les ticks sont également approximatifs puisqu'ils dépendent des autres activités et du chargement en mémoire de FS .

Les valeurs entre 1 et 59 millisecondes sont actuellement maintenues par un fil séparé dans FSUIPC4 et donnent des résultats plus précis , mais notez que sonder trop fréquemment les joysticks peut diminuer les performances de FS , et même rendre plus précaires sa réponse aux commandes de joystick . On n'a pas noté de véritables effets contraires durant le testing , mais vous êtes avertis . Si vous pensez que vous avez besoin de sonder plus rapidement un Bouton , essayez des valeurs dans la gamme 10-25, et assurez-vous à chaque fois que FS fonctionne efficacement .

Notez que les joystick émulés PFCFSX (ceux avec des nombres supérieurs à 16) sont sondés 4 fois plus fréquemment , quelque soit le cas – parce qu'il n'y a pas de dépassement (overhead) à le faire – il n'y a pas d'appel à Windows , mais simplement des inspections de données . Les boutons GoFlights (nombres joystick encore plus haut) ne sont pas sondés du tout- FSUIPC4 reçoit un appel du pilote interface GoFlight (GFDev.DLL) selon qu'un événement arrive .

FORMATS DE DEFINITION DE BOUTONS

La programmation de Boutons est sauvegardée dans des sections du fichier .INI . Pour les boutons globalement en vigueur , c'est dans [Buttons] . Pour les boutons avion-spécifique , c'est dans [Buttons.<aircraft name>] . Dans chaque section peuvent être inclues jusqu'à 2048 entrées séparées , définissant autant d'actions de boutons , numérotées normalement séquentiellement à partir de 0 , en sorte que le total des définitions dans la section Global et la section la plus grande avion-spécifique ne soit pas plus grande que 2048 .

Si le paramètre [General] **ShortAircraftNameOk** est réglé sur **Yes** ou **Substring** , la partie <aircraft name> de l'en-tête de section peut être abrégée (manuellement , en éditant le fichier .INI) de telle sorte que ça s'applique à plus d'un avion . Avec l'option « Yes » , FSUIPC4 sélectionnera automatiquement la section avec l'appariement le plus long . L'ordre des sections dans le fichier .INI n'est pas pertinent . Cependant , avec l'option « Substring » , la première section avec un appariement substring sera sélectionnée – pas de concept de « plus grand appariement » dans ce cas .

Le format basique de chaque entrée dans la section Buttons est le suivant :

Pour des appuis touches de clavier : <Entry number> = <Action><Joy#>,<Btn#>,K<key>,<shifts>

Pour des commandes : <Entry number> = <Action><Joy#>,<Btn#>,C<control>,<parameter>

Pour des macros (voir la section séparée sur les macros) : <Entry number> = <Action><Joy#>,<Btn#>,M<file#> :<ref#>,<parameter>

Le format des paramètres devient plus complexe pour des actions conditionnelles , ils seront décrits plus tard .

Le <Entry number> n'est pas matériel la plupart du temps – excepté dans les séquences pression-relâchement de boutons uniques . C'est juste un nombre séquentiel de 0 à 2047 (mais limité à un total de 2048 entrées pour la section générale plus n'importe quelle section Avion-spécifique) .

Chaque entrée doit avoir un numéro d'entrée unique , et l'ordre actuel n'est important que si de multiples actions sont définies pour le même Bouton . FSUIPC4 conservera la numérotation , d'où l'ordre est défini par le numéro (pas la position sur la ligne)

Vous pouvez ajouter des commentaires séparés par des points virgule (;) à la fin de la ligne , et ceux-ci seront conservés . Vous pouvez aussi insérer des lignes qui ne contiennent que des commentaires , mais elles ont besoin aussi d'un <Entry number> , d'un autre côté elles peuvent ne pas conserver leur position relative . Des commentaires peuvent contenir jusqu'à 63 caractères – les plus longs seront tronqués si et quand la section [Buttons] sera réécrite par FSUIPC4

<Action> est une simple lettre indiquant l'action en train d'être définie :

P Pulse l'appui touche clavier ou la commande : c'est-à-dire ne laisse pas la touche appuyée pendant que le bouton est maintenu appuyé . C'est toujours le cas pour les commandes ,et ça devrait l'être pour les appuis touches clavier qui utilisent ALT, parce qu'une fois que le menu FS est entré , FSUIPC4 ne peut plus fournir des changements ultérieurs comme des relâchements de touches

H maintient en appui les touches spécifiées jusqu'à ce que le Bouton soit relâché . (ça ne s'applique pas aux commandes , qui seront , dans

ce cas , traitées comme P) N'utilisez pas ceci avec les appuis touches clavier qui impliquent ALT , pour la raison déjà donnée

R répète l'appui touche clavier ou la commande pendant que le Bouton est gardé appuyé . Le taux de répétition , non ajustable , est d'environ 6 par seconde . N'utilisez pas ceci avec les appuis touches clavier qui impliquent ALT , pour la raison déjà donnée

U pulse l'appui touche clavier ou la commande lorsque le Bouton est relâché .

N'importe quel Bouton peut avoir une entrée U , P , H ou R . Qu'il stipule que le Bouton n'a qu'une seule entrée , P, H ou R et/ou U, et que lorsqu'il en a deux elles consistent soit toutes les deux en appuis touches clavier ou toutes les deux en commandes , la programmation des Boutons peut être effectuée entièrement dans la page Buttons de FSUIPC4 .

Le <Joy#> identifie le numéro de joystick (0-15 pour les joysticks ordinaires , 16 et au-dessus pour les PFC , GoFlight ou autres futurs joysticks « émulés ») ., ainsi que l'affiche FSUIPC4 ; et le <Btn#> identifie le Bouton spécifique (0-39) , de la même façon . Parmi ces Boutons , 0-31 sont les Boutons habituels et 32-39 sont les 8 angles de vues possibles POV, démarrant en avant et tournant dans le sens des aiguilles d'une montre tous les 45 degrés .(il n'y a pas de POV émulés , c'est pourquoi , pour les joysticks 16 et au-dessus , les numéros de Boutons sont toujours dans la gamme 0-31 .

Notez que les numéros de joysticks peuvent être remplacés par une lettre assignée (A-Z en omettant I et O) si la fonctionnalité JoyNames est utilisée pour assigner des joysticks indirectement , au cas où leur numéro ID réel changerait .

Lorsque les Boutons sont programmés sur WideFs clients , le numéro de joystick inclut également un numéro de Client PC – 1000 pour le Client 1 , 2000 pour le Client 2 et ainsi de suite . La numérotation Client est actuellement mise en œuvre par WideServer , qui garde un enregistrement des noms de Client PC et leur assigne des numéros dans le fichier WideServer.ini . Vous devrez seulement vous en soucier lorsque vous changerez de PC ou que vous voudrez les renommer .

Pour les appuis touches clavier , la valeur <key< qui suit la lettre « K » est le code key virtuel pour les appuis touches clavier . Voici une liste ,

pour commodité (mais notez que tous ces codes ne seront pas utilisables) :

0 Nul (+Alt , Shift etc. seuls)
(peut seulement être utilisé pour appuyer sur elles , pas pour les détecter !)

8 Backspace
12 Pavé numérique 5 (Numlock sur OFF)
13 Enter
16 Shift (nécessite une valeur de Shift 9)
17 Ctrl (nécessite une valeur de shift 10)
18 Menu (nécessite une valeur de shift 72)
19 Pause
20 CapsLock
27 Escape
32 barre espace
33 page up
34 page down
35 End
36 Home
37 Left arrow
38 Up arrow
39 Right arrow
40 Down arrow
44 Print Screen
45 Insert
46 Delete
48 0 sur clavier principal
49 1 sur clavier principal
50 2 sur clavier principal
51 3 sur clavier principal
52 4 sur clavier principal
53 5 sur clavier principal
54 6 sur clavier principal
55 7 sur clavier principal
56 8 sur clavier principal
57 9 sur clavier principal
65 A
66 B
67 C
68 D
69 E

70 F
71 G
72 H
73 I
74 J
75 K
76 L
77 M
78 N
79 O
80 P
81 Q
82 R
83 S
84 T
85 U
86 V
87 W
88 X
89 Y
90 Z
96 Pavé numérique 0 (NumLock ON)
97 Pavé numérique 1 (NumLock ON)
98 Pavé numérique 2 (NumLock ON)
99 Pavé numérique 3 (NumLock ON)
100 Pavé numérique 4 (NumLock ON)
101 Pavé numérique 5 (NumLock ON)
102 Pavé numérique 6 (NumLock ON)
103 Pavé numérique 7 (NumLock ON)
104 pavé numérique 8 (NumLock ON)
105 Pavé numérique 9 (NumLock ON)
106 Pavé numérique *
107 Pavé numérique +
109 Pavé numérique –
110 Pavé numérique .
111 Pavé numérique /
112 F1
113 F2
114 F3
115 F4
116 F5
117 F6
118 F7

119 F8
120 F9
121 F10
122 F11
123 F12
124 F13
125 F14
126 F15
127 F16
128 F17
129 F18
130 F19
131 F20
132 F21
133 F22
134 F23
135 Pavé numérique Enter (ou F24)
144 NumLock
145 ScrollLock
186 touche * ; :
187 touche * = +
188 touche * , <
189 touche * - _
190 touche * . >
191 touche * / ?
192 touche * ‘ @
219 touche * [{
220 touche * \ |
221 touche *] }
222 touche * # ~~
223 touche * ¬

* ces touches varieront de clavier à clavier . Les graphismes indiqués sont ceux affichés par mon clavier UK . Il est possible que ces touches , dans la même position relative sur le clavier répondront de façon similaire , il s’agit donc d’une description positionnelle pour les utilisateurs de claviers non UK . Cette liste est , de gauche à droite et de haut en bas :

223 , 189 , 187 , 219 , 221 , 186 , 192 , 222 , 220 , 188 , 190 , 191

La valeur <shifts> est une combinaison (ajoutez les) des valeurs suivantes , selon les besoins :

1 Shift
2 Control
4 Tab
8 Normal (à ajouter dans tous les cas)
16 Alt (faites attention à celui-là , ça appelle le Menu)
32 touche Windows (gauche ou droite)
64 touche Menu (la touche application , à droite de la touche Windows droite)

[*Notez que les touches Tab et Alt sont indiqués ici par bits , à l'opposé de ce qui est utilisé pour programmer les touches . Je m'en excuse , car il est trop tard pour changer cet oubli de conception*]

Si vous avez besoin seulement de « normal » le paramètre entier et la virgule précédente peuvent être oubliés . Les valeurs usuelles sont :

9 pour Shift+
10 pour Control+
11 Pour Shift+Control+

Pour les commandes FS , le <control> est un nombre à partir de 65536 et au-dessus , indiquant le numéro de commande spécifique FS. Vous pouvez en trouver des listes dans mes documents de commandes FS . dans la page Buttons de FSUIPC4 , les commandes sont affichées normalement par noms , mais si vous voulez essayer une commande qui n'a pas de nom mais qui pourrait faire quelque chose d'utile pour vous , entrez ici , dans le fichier .INI . Dans la page Buttons , FSUIPC4 l'affichera par numéro à la place des noms .

Le <parameter> pour une commande est optionnel – omettez-le ainsi que la virgule précédente pour la plupart des commandes de type interrupteur à bascule/bouton . Une valeur de paramètre de 0 sera adoptée de toute façon .

L'une ou l'autre ou les deux valeurs <control> et <parameter> peuvent être données en hexadécimal , précédées par un caractère « x » .

Tout comme les commandes FS , un nombre de commandes supplémentaires FSUIPC4 sont ici disponibles .Leur gamme , de 1000 à 3000 et leurs valeurs « xcc00zzzz » (en hexadécimal) encodent les commandes « offset » de FSUIPC4 . Voir la liste en-dessous de la rubrique « keys » pour plus de détails .

SEQUENCES , COMBINAISONS ET MIXTURES

Le page Buttons de FSUIPC4 est délibérément maintenue pour offrir des choix simples , dissimulant quelques unes des possibilités de programmation . En éditant le fichier .INI , vous pouvez faire davantage :

- maintenir un appui touche clavier tout en pressant une autre touche
- presser et relâcher une séquence de touches clavier
- mélanger appui touches clavier et commandes FS dans le fonctionnement d'un seul Bouton
- rendre l'action de boutons conditionnelle à l'état d'autres boutons (voir « compound » boutons (composés) , ci-dessous

Les 3 premiers sont faits simplement en définissant les actions dans des entrées séparées , chacune référant au même numéro de joystick/bouton . Je vous ai recommandé d'utiliser d'abord la page Buttons pour obtenir l'action programmée initiale (ce qui vous assure d'avoir le bon numéro de bouton) , puis fermez FS et éditez les entrées déjà faites dans le fichier .INI . La seule chose importante est de numéroter séquentiellement les entrées – de préférence consécutivement , mais pas nécessairement .

Exemples :

16=H1,2,K69,8
17=H1,2, K49,8

Appuyez sur la touche « E » et maintenez l'appui , puis pressez et maintenez l'appui sur la touche « 1 » , de telle sorte que vous appuyez sur les deux ensemble. Elles sont toutes les deux relâchées (dans le même ordre) lorsque le bouton est relâché .

18=P1,3,K69,8
19=P1,3,K49,8
20=P1,3,K50,8
21=P1,3,K51,8
22=P1,3,K52,8

Pressez et relâchez “E” , puis “1” , “2” , “3” et “4” en succession rapide , pour sélectionner tous les Moteurs .
23=P2,3,K76,24
24=P2,3,K65,8
25=P2,3,K69,8

Presser et relâcher Alt+L puis A puis E est une succession très rapide . FSUIPC ne laisse aucun délai entre les actions lorsque la touche ALT est utilisée . D’un autre côté , dès qu’il permet à la procédure des touches de commencer , la combinaison de touches ALT va amener l’item menu et (dans ce cas) la boite de dialogue, FSUIPC ne se lancera pas et sera donc incapable de fournir le relâchement des touches . D’horribles mélanges peuvent s’ensuivre ! …

Le dernier exemple que j’utilise actuellement est réel . ALT+L fait venir le menu Lago , « A » sélectionne FSAssist et « E » le repoussage (pushback) avec Engine start . ça vous met dans la boite de dialogue pushback , mais alors j’ai bien peur que vous soyez en train d’utiliser la souris . FSUIPC ne peut pas vous aider davantage .

CONDITIONS BOUTON COMPOSE (COMPOUND)

Des fonctionnalités sont inclues pour vous permettre de spécifier des actions pour un bouton , qui sont dépendantes de l’état d’un autre bouton (ou mieux , d’un interrupteur) . ça se fait en utilisant ce que j’appelle la programmation de bouton « composé » - quoiqu’elle pourrait être également « conditionnelle » ou « coopérative » . Quoiqu’il en soit , j’utilise la lettre « C » dans les définitions , comme suit :

n=CP(+j2,b2)j,b,…
n=CU(+j2,b2)j,b,…
n=CP(-j2,b2)j,b,…
n=CU(-j2,b2)j,b,…

où « C » indique le checking du Bouton composé , P = pulse) la pression , U = pulse au relâchement. Vous pouvez aussi utiliser CR à la place de CP pour répéter une action – la répétition continue tant que toutes les conditions sont « vraies » .Il n’y a pas de fonctionnalité pour

l’action de maintenir appuyé (hold) pour ce qui concerne les boutons composés .

L’intérieur des parenthèses contient des détails sur le bouton secondaire , qui doit être dans certaines conditions pour que le bouton courant fonctionne :

(+j2,b2) signifie que le bouton b2 sur le joystick j2doit être pressé (« on ») pour que l’action du bouton courant (pour j,b,) soit autorisée .

(-j2,b2) signifie que le bouton b2 sur le joystick j2doit être relâché (« off ») pour que l’action du bouton courant (pour j,b,) soit autorisée .

La partie j,b, est la paramètre usuel du bouton pour effectuer une action à partir du bouton « courant » , c’est-à-dire le bouton b sur le joystick j .

Vous pouvez avoir une condition , comme affiché ci-dessus , ou deux , ou plus (jusqu’à 16 , en fait) , comme ceci :

n=CP(+j2,b2)(+j3,b3)j,b,…

où , désormais , les deux conditions entre parenthèses à la fois doivent être réunies pour que l’action du bouton : j,b, s’inscrive dans l’événement tel qu’il a été défini .

Les conditions peuvent être créées pour s’appliquer non pas à l’état actuel d’un bouton , mais à l’état d’un « drapeau » (« flag ») , installé et dégagé par un bouton (ou même par un appui touche clavier) . Pour chaque bouton « normal » possible (16 joysticks x 32 boutons = 512 boutons) FSUIPC4 met en œuvre (maintains) un drapeau (F pour « flag ») . Chaque fois qu’un bouton est pressé (passe de off à on) , FSUIPC4 bascule (toggles) son drapeau . ça fait des boutons « flags » des sortes d’interrupteurs « à loquet » (latching) . Vous pouvez le tester dans toute condition entre parenthèses , qui précède la condition par F , donc :

N=CP(F+j2,b2) …

Ça dit que le reste de ce paramètre est respecté (obeyed) si le Drapeau associé à j2,b2 est réglé (set) . La condition (F-j2,b2) teste pour que le Drapeau soit dégagé. Notez que l’état actuel courant du bouton j2,b2 n’est pas pertinent . Tout ceci selon que le Drapeau est installé ou dégagé .

Dans un réglage multi-conditionnel , n'importe quelle condition peut s'appliquer aux Drapeaux .
Ces boutons Flags peuvent également être installés , effacés et basculés (toggled) par 3 commandes spéciales FS , **Button Flag Set** (C1003) , **Button Flag Clear** (C 1004) et **Button Flag Toggle** (C 1005) . dans ces 3 cas , le joystick (0-15 seulement) et le bouton (0-31) sont donnés dans le paramètre par une valeur calculée comme suit :

256 * J + B (par exemple joystick 15 et bouton 31 donneraient 3871)

Ces 3 commandes sont listées dans le menu déroulant dans les pages d'assignement Buttons & Keys , donc vous pouvez les programmer là , ou dans le fichier .INI . Avec ces commandes résultant de l'action conditionnelle de boutons , vous pouvez influencer les conditions d'action des boutons dans une multitude de voies .

Un point à noter : depuis que vous pouvez utiliser le clavier ou autre bouton composé pour installer , effacer ou basculer les Drapeaux , le bouton actuel pour lequel le Drapeau est assigné n'a pas besoin d'exister actuellement !

Ok . Maintenant , qu'est-ce que ça veut dire réellement ? …Quelques exemples simples suffiront ici . Je laisse venir les plus imaginatifs d'entre vous avec des applications réellement complexes .

Premièrement , ça veut dire que vous pouvez assigner de multiples utilisations à n'importe quel nombre de boutons en les rendant conditionnels à un nombre d'autres boutons .Par exemple un interrupteur à bouton rotatif à loquet à 12 positions peut être connecté pour faire fonctionner les boutons 1 à 12 sur le joystick1 . Puis , pour n'importe quel autre bouton , je peux programmer 12 actions différentes . par exemple , bouton0,3 pourrait avoir 12 actions différentes assignées , comme ceci :

1=CP(+1,1)0,3, …
2=CP(+1,2)0,3, …
3=CP(+1,3)0,3, …
…
12=CP(+1,12)0,3, …

Etc … Par exemple vous pouvez avoir un réglage d'assignements pour des opérations au sol , un pour le décollage , un pour la montée , un pour la croisière , etc …

Deuxièmement , de sorte à économiser effectivement sur l'utilisation des boutons , là où vous avez besoin d'un interrupteur à bascule , vous pouvez faire en sorte qu'un bouton bascule (toggle) entre deux actions en utilisant un Drapeau comme une condition . Par exemple , supposons que votre bouton est : joy11, bouton3et un Drapeau de rechange (spare flag) est : 15,2 (un bouton non utilisé sur les joysticks 0-15) . Programmez votre bouton en 3 lignes sur FSUIPC , les nombres à gauche doivent être séquentiels avec tout ce qui est déjà là , mais je supposerai que vous n'en avez pas d'autres , donc : démarrons avec 1)

1=P11,3,C1005,3842

Ça dit : exécutez la commande 1005 chaque fois que votre bouton est pressé . Control 1005 est « Button Flag Toggle ». La paramètre 3842 identifie le Drapeau : 256xjoystick15+bouton2. Ce Drapeau alternera donc désormais entre installé et effacé chaque fois que vous appuierez sur le bouton .

2=CP(F+15,2)11,3, …

Ça dit quoi faire à FSUIPC si le bouton est pressé ET que le Drapeau est installé . Remplacez les points de suspension par le numéro de commande et le paramètre pour une des actions dont vous avez besoin

3=CP(F-15,2)11,3, …

De la même façon , ça dit ce que fait FSUIPC lorsque le bouton est pressé et que le Drapeau n'est pas installé .

Troisièmement , vous pouvez maintenant programmer ces interrupteurs rotatifs de type 2 phases , les unes où tourner le cran dans un sens donne des pulsations sur les deux lignes de phase embrayées une fois , et tourner d'un cran dans l'autre sens donne la relation de phase opposée .

Disons que les entrées qui viennent de l'interrupteur rotatif sont sur Joystick1 , boutons 1 et 2 . Lorsque B1 est ON et que B2 passe de off à on , alors l'axe a tourné d'un cran . Quand B1 est ON et B2 passe de on

à off , l'axe a tourné d'un cran dans l'autre sens . C'est l'exemple le plus simple :

1=CP(+1,1)1,2, … tourner direction 1 action
2=CU(+1,1)1,2, … tourner direction 2 action

Vous pouvez aussi avoir une action à double vitesse , fonctionnant à chaque changement sur B2 , de on à off et de off à on . Ajoutez juste deux conditions supplémentaires :

3=CP(-1,1)1,2, … tourner direction 2 action (B2 de off à on quand B1 est off)
4=CU(-1,1)1,2, … tourner direction 1 action (B2 de on à off quand B1 est off)

Puisque le chose tourte entière est complètement symétrique (il n'y a aucune raison que B1 commande B2, ça pourrait aussi bien être le contraire) vous pouvez le programmer actuellement pour agir sur tous les bords (edges) des deux boutons , en ajoutant 4 autres conditions :

5=CP(+1,2)1,1, … tourner direction 2 action (B1 de off à on quand B2 est on)
6=CU(+1,2)1,1, … tourner direction 1 action (B1 de on à off quand B2 est on)
7=CP(-1,2)1,1, … tourner direction 1action (B1 de off à on quand B2 est off)
8=CU(-1,2)1,1, … tourner direction 2 action (B2 de on à off quand B2 est off)

Donc vous pouvez effectivement choisir combien de pulsations vous obtiendrez pour un taux de rotation donné (turning rate) . Comme vous pouvez le voir , vous pouvez obtenir des taux de 1x , 2x ou 4x - même 3x si vous faites une partie pour seulement la moitié des changements ! Notez que pour la fiabilité à hautes vitesses , vous pouvez avoir besoin de réduite le **PollInterval** .

A propos , c'est avec certains de ces interrupteurs rotatifs que la double condition peut devenir très utile . Si vous avez un bouton rotatif simple de ce type , avec aussi une action de poussoir disponible , vous pouvez le programmer pour ajuster à la fois les unités et les fractions disons d'une fréquence radio . Utilisez simplement le Drapeau associé à l'action

du bouton pour choisir entre un couple d'actions ou un autre , donc , en supposant que le bouton est 1,3 :

1=CP(F+1,3)(+1,1)1,2, …augmenter fraction
2=CU(F+1,3)(+1,1)1,2, … diminuer fraction
3=CP(F-1,3)(+1,1)1,2, … augmenter entier
4=CU(F-1,3)(+1,1)1,2, … diminuer entier

Une dernière chose . Si vous utilisez plusieurs boutons rotatifs de ce type (c'est-à-dire avec les 2 signaux dans différentes phases de relation pour indiquer la direction de la rotation) s'ils sont du type à avoir les 2 signaux off dans la détente vous pouvez sauvegarder les connections du bouton en rendant l'un deux commun (sur chacun) . Si vous le faites , vous ne pourrez tourner qu'un d'entre eux à la fois , mais c'est probablement une restriction qui vaut la peine si vous êtes à court de connections de bouton .

AJOUTER DES CONDITIONS OFFSET

Tout comme ce qui est au-dessus (et au-dessous pour Keys) n'iimporte laquelle ou toutes les entrées de toutes les sections Buttons & Keys de FSUIPC4.INI peuvent chacune contenir une condition unique basée sur la valeur de bits , bytes , words ou double-words dans l'interface FSUIPC4 IPC . Ces valeurs sont adressées par une valeur « offset » en hexadécimal et inclut tout ce que vous pourrez penser au sujet de tout ce qui peut arriver dans FS .

Prenons quelques exemples . Vous pouvez créer des conditions basées sur :

- avion en vol ou au sol
- moteurs en marche ou pas
- une lumière spécifique ou plus sur on ou off
- train d'atterrissage levé ou rentré
- et même signaux radio valides pour NAV1 , NAV2 , GS , ILS , LOC …etc … les possibilités sont infinies ! …

Pour faire un bon usage de ceci , vous aurez besoin du Guide du Programmeur , qui liste tous les offsets . Ce document est dans le FSUIPC4.SDK. Vous y trouverez quantité de données dont vous ne pourrez pas vous servir – ici , les conditions impliquent de traiter avec

des bits ou des valeurs en bytes 8-bit , des words en 126-bit et des double-words en -32 bit . Vous ne pouvez pas utiliser de valeurs « string » , « table » ou « floating point » .

Vous pouvez ajouter une condition offset sur n'importe quelle ligne paramètre Buttons ou Keys de FSUIPC4.INI comme suit :

<sequence number>=<offsetcondition> <usual parameter>

L'espace entre la nouvelle condition et le paramètre usuel est essentiel . Un simple exemple va nous aider . prenez le bouton paramètre push , censé basculer le train d'atterrissage lorsque le bouton est poussé :

1=P1,0,C65570,0

En ajoutant une condition offset , nous pouvons stopper ceci lorsque l'avion est au sol :

1=W0366=0 P1,0,C65570,0

L'insertion « W0366=0 » spécifie que le Word (valeur 16-bit ou 2-bit) à l'offset 0366 doit être à zéro pour que cette ligne soit obéie (to be obeyed) . L'Offset 0366 contient 0 quand l'avion est en l'air , 1 lorsqu'il est au sol .

Le format de la condition est :

<size> <offset> <mask> <condition>

Où

<size> est B our Byte , W pour Word et D pour Double Word

<offset> représente l'offset FSUIPC , une valeur hexadécimale comprise en 0000 et FFFF

<mask> est optionnel , et s'il est donné , il sélectionne un ou plusieurs bits : spécifié comme &x où « x » représente le mask 8 , 16 ou 32-bit en hexadécimal . la valeur dans l'offset est « ANDée » avec ce mask avant d'être utilisée .

<condition> est soit :

= valeur pour une égalité

! valeur pour une inégalité
< valeur pour inférieur à
> valeur pour supérieur à

Et ici , la « valeur » est décimale tant qu'elle n'est pas précédée par un x (ou un X) auquel cas elle est hexadécimale , comme offset et mask . FSUIPC sortira de l'hexadécimal quand un mask est utilisé , sinon du décimal . Toutes les valeurs sont traitées comme non signées .

Le mask optionnel est utile pour tester des bits spécifiques , comme dans le cas des interrupteurs lumière dans l'offset 0D0C ou les détails de réception radio dans l'offset 3300 . Par exemple , la condition offset :

W3300&0040 !0

Est VRAIE quand la NAV1 couramment réglée (tuned) est pour un ILS .

La partie <condition> est également optionnelle , réglée à !0 par défaut , ainsi , ce dernier exemple pourrait être abrégé en :

W3300&0040

Pour les Utilisateurs de Projet Magenta (PM) qui utilisent quelques fois le pilote automatique par défaut de FS , une condition très utile est simplement :

W0500

L'Offset 0500 est non-zéro quand le MCP de PM fonctionne, sinon il est à 0 , vous pouvez donc programmer boutons et appuis touches clavier pour faire fonctionner PM, sinon FS fonctionnera .

Finalement , pour un switching plus ingénieux , vous pouvez vouloir utiliser un bouton pour ajuster une valeur offset FSUIPC4 qui , alors , via des conditions offset , sélectionnera entre des assignements d' un numéro de bouton alternatif et/ou d'un appui touche clavier .Pour ce faire , les offsets 66C0 à 66FF sont réservés juste pour vous pour faire ce que vous voulez . Les commandes offsets cycliques augmenter/diminuer permettent , disons , à une valeur byte de circuler à travers un nombre de valeurs , chaque valeur sélectionnant des actions particulières pour des boutons ou des appuis touches clavier définis . Les entrées dans Buttons ou Keys pourraient ressembler à ceci :

31=P174,10,Cx510066C0,x00030001
32=B66C0=0P117,6,C1030,0
33=B66C0=1P117,6,C1034,0
34=B66C0=2P117,6,C1038,0
35=B66C0=3P117,6,C1042,0

Ici la valeur dans le Byte à l'Offset 66C0 est parcourue (cycled) de 0 à 3 et de 3 à 0 , par le bouton 174,10 , et cette valeur , à tour de rôle , sélectionne ce qui arrive avec le bouton 117,6 .

Ce sont des exemples réels relatifs à la programmation d'une unité GoFlight GF-45 pour différents ajustements de fréquence . Il y aura plein d'exemples complets dans la documentation de mon programme GF display , attendu sous peu (due shortly) .

POUR ALLER PLUS LOIN , POUR DE MEILLEURS EXEMPLES

Des exemples supplémentaires , intéressants et utiles de programmation de bouton sont fournis en Annexe du présent document . L'annexe a été gracieusement fournie par un utilisateur enthousiaste de FSUIPC , à qui je suis redevable .

ERREURS DANS LES PARAMETRES DE BOUTON

Lorsque les sections [Buttons] sont lues (ou relues via le bouton « reload » dans la page Buttons de FSUIPC4) , les lignes sont vérifiées minutieusement . Celles qui sont syntaxiquement fausses sont ignorées . Cependant , lorsqu'une ligne est ignorée , un message d'erreur est apposé sous la forme :

…<<ERRORn …

Les numéros d'erreurs possibles sont listés ci-dessous . Vous pouvez alors corriger la ligne et presser à nouveau sur « reload » pour la revérifier . Vous ne devez pas gommer (erase) les additions d' <<ERROR … Si la ligne est OK , le message sera gommé . S'il y a encore une erreur , un nouveau numéro d'erreur apparaîtra .

Les erreurs sont :

1 condition Offset : pas d'offset hexadécimal suivant la taille (B,W ou D)
2 condition Offset : l'offset est trop grand (plus de 4 digits hex)
3 condition Offset : la partie « &mask » n'a pas de mask hexadécimal
4 condition Offset : le mask est trop grand (plus de 8 digits hex)
5 condition Offset : condition non reconnue (pas de = , ! , < , > , ou « espace » représentant !0)
6 condition Offset : valeur de comparaison X pour hex non suivie par une valeur hex
7 condition Offset : valeur de comparaison X pour hex trop grande (plus de 8 digits hex)
8 condition Offset : pas de valeur décimale ou Xhex après = , ! , < ou >
9 opération de Bouton non spécifiée , telle que H , P , R , U ou C
10 opération de Bouton conditionnelle , pas de P , R ou de U après le C
11 trop de (…) conditions de Bouton
12 condition numéro Joystick trop grand
13 numéro de Bouton omis dans condition (le ,b dans (j,b))
14 pas de) pour condition (
15 le numéro de Bouton ne peut pas être >31 dans la condition
16 le bouton principal du numéro de Joystick est trop grand
17 le numéro de Bouton principal est supérieur à 39
18 virgule (,) manquante après le numéro de bouton principal
19 le C.r Kor M nécessaire pour Control , Key ou Macro est manquant
20 format inconnu , syntaxe inintelligible .

PROGRAMMATION TOUCHES CLAVIER

La boite de dialogue d'options FSUIPC4 fournit une page pour programmer des appuis touches clavier pour assigner des commandes uniques à FS . Nous examinons ici comment cette programmation est encodée dans le fichier FSUIPC4.INI , et comment la programmation peut être tendue pour fournir de commandes multiples à partir d'une simple combinaison de touches .

FORMAT DE DEFINITION DE TOUCHES

La programmation de touches est sauvegardée dans des sections du fichier .INI . , pour les touches globalement opérationnelles , elle s'appelle [Keys] , pour les boutons avion-spécifique , [Keys ,<aircraft name>]Peuvent être inclues dans chaque section jusqu'à 1024 entrées séparées définissant des actions de touches , numérotée séquentiellement à partir de 0 , indiquant que le total des définitions dans la section Global et dans la plus grande section avion-spécifique n'est pas supérieur à 1024 .

Tout comme les paramètres Buttons , les entrées appuis touches clavier sont rechargées en mémoire chaque fois que vous changez d'avion dans FS , donc vous pouvez faire des changements dans le fichier .INI et les tester sans recharger FS en mémoire .

Si le paramètre [General] **ShortAircraftName OK** est réglé sur **Yes** ou **Substring** , la partie <aircraftname> de la section d'en-tête peut être abrégé (manuellement , en éditant le fichier .INI) de telle sorte que ça s'applique à plus d'un avion . Avec l'option « Yes » , FSUIPC4 sélectionnera automatiquement la section avec l'appariement le plus long (the longest match) . L'ordre des sections dans le fichier .INI n'est pas pertinent . Cependant , l'option « Substring » sélectionnera la première section avec un appariement substring- il n'y a pas de concept d'appariement le plus long dans ce cas .

Le format de chaque entrée dans la section Keys apparaît comme suit :

n=key,shifts,control, parameter

Pour seulement un appui touche clavier , ou

n=key,shifts,control1, parameter1,control2,parameter2

pour une touché avec pression (1) et relâchement (2) .

Ici ,

n va de 0 à 1023 (c'est-à-dire qu'au maximum , 1024 différentes actions d'appui touches clavier peuvent être ajoutées) .

key codekey virtuel , comme dans le fichier FS.CFG (voir liste ci-dessus dans la section sur les Buttons)
Note : si les répétitions automatiques d'appuis touches clavier doivent être ignorées , ce code est précédé par la lettre « **N** » .

Shifs 8 normal
+1 shift
+2 control
+4 Alt (en réalité pas très utile)
+16 Tab (un « shift » supplémentaire pour donner davantage de combinaisons)
+32 touche Windows (gauche ou droite)
+64 touche Menu (la touche application , à droite de la touche Windows droite)

[*Notez que les touches Tab et Alt sont ici indiquées par bits à l'opposé de leur utilisation pour la programmation de boutons . Je m'en excuse , c'est un bug de conception , trop tard désormais à modifier*]

Control c'est normalement un numéro de commande FS (comme dans mes listes) ou un numéro spécial FSUIPC4 pour des commandes supplémentaires . Il peut être en décimal , ou bien , s'il est précédé par un « x » en hexadécimal . Les commandes supplémentaires FSUIPC4 vont de 1000 à 3000 et xcc00zzzz en hexadécimal qui encode les commandes offset FSUIPC . Voir la liste ci-dessous pour plus de détails .

Alternativement , ce peut être une référence Macro qui , dans ce cas , est de forme :

M <file#> :<ref#>

Par exemple M3 :4 se réfère à macro file3 , macro numéro 4 dans ce fichier . Voyez s'il vous plaît la section sur les Macros pour davantage de détails .

Ça peut aussi être une référence Lua plug-in :

L<file#> :<action>

Où le numéro Fichier réfère à la liste [Lua File] dans le fichier .INI , et l'action est une de ces lettres :

R=Run,K=Kill,S=Set,C=Clear,T=Toggle,D=Debug

Parameter valeur pour aller avec control , pour des types "Set" ou quelques commandes spéciales FSUIPC . Normalement en décimal , mais en hexadécimal si précédé d'un « x »

ERREURS DANS LES PARAMETRES DES TOUCHES

Lorsque les sections [Keys] sont lues (ou relues via le bouton « reload » dans la page Buttons de FSUIPC4) , les lignes sont vérifiées minutieusement . Celles qui sont syntaxiquement fausses sont ignorées . Cependant , lorsqu'une ligne est ignorée , un message d'erreur est apposé sous la forme :

…<<ERRORn …

Les numéros d'erreurs possibles sont listés ci-dessous . Vous pouvez alors corriger la ligne et presser à nouveau sur « reload » pour la revérifier . Vous ne devez pas gommer (erase) les additions d' <<ERROR … Si la ligne est OK , le message sera gommé . S'il y a encore une erreur , un nouveau numéro d'erreur apparaîtra .

Les erreurs sont :

1 condition Offset : pas d'offset hexadécimal suivant la taille (B,W ou D)
2 condition Offset : l'offset est trop grand (plus de 4 digits hex)
3 condition Offset : la partie « &mask » n'a pas de mask hexadécimal

4 condition Offset : le mask est trop grand (plus de 8 digits hex)
5 condition Offset : condition non reconnue (pas de = , ! , < , > , ou « espace » représentant !0)
6 condition Offset : valeur de comparaison X pour hex non suivie par une valeur hex
7 condition Offset : valeur de comparaison X pour hex trop grande (plus de 8 digits hex)
8 condition Offset : pas de valeur décimale ou Xhex après = , ! , < ou >
20 format inconnu , syntaxe inintelligible .
21 numéro touche virtuel pas dans la gamme 1-255
22 pas de virgule (,) après numéro de touche
23 pas de virgule (,) après valeur code shift
24 mauvaise valeur control

COMMANDES SUPPLEMENTAIRES « FS » AJOUTEES PAR FSUIPC4

1001	PTT on (pour Sqawbox3 , Roger Wilco ou AVC Advanced Voice Client)
1002	PTT off (pour Sqawbox3 , Roger Wilco ou AVC Advanced Voice Client)
1003	Réglage bouton flag (param=256*joy+btn ouJjBb)
1004	Effacer bouton flag (param=256*joy+btn ouJjBb)
1005	Basculer (toggle) bouton flag (param=256*joy+btn ouJjBb)
1006	Envoi touche (keysend) vers WideClients (param=Keysend number , 1-255)
1007	Réglage Autobrake (param=0 pour RTO , 1=off,2-5 pour 1,2,3,max)
1008	Réglage Densité Trafic (param=0-100%)
1009	Bascule (toggle) Densité Trafic (param=0-100%)
1010	Volet augmentation (par 512 ou réglage quantité dans SpoilerIncrement=INI parameter)
1011	Volet diminution (par 512 ou réglage quantité dans SpoilerIncrement=INI parameter)
1012	…
1013	…
1014	…
1015	…
1016	Ap Alt Var Dimin rapide (-1000)

1017	Ap Alt Var Augm rapide (+1000)
1018	Ap Mach Var Dimin rapide (-10)
1019	Ap Mach Var Augm Rapide (+10)
1020	Ap Spd Var Dimin rapide (-10)
1021	Ap Spd Var Augm rapide (+10)
1022	Ap Vs Var Dimin rapide (-1000)
1023	Ap Vs Var Augm rapide (+1000)
1024	HeadingBug Dimin rapide (-10)
1025	Heading Bug Augm rapide (+10)
1026	Vor1 Obi Dimin rapide (610)
1027	Vor1 Obi Augm rapide (+10)
1028	Vor2 Obi Dimin rapide (610)
1029	Vor2 Obi Augm rapide (+10)
1030	Utilisation Com1 entier augm
1031	Utilisation Com1 entier dimin
1032	Utilisation Com1 frac augm
1033	Utilisation Com1 frac dimin
1034	Utilisation Com2 entier augm
1035	Utilisation Com2 entier dimin
1036	Utilisation Com2 frac augm
1037	Utilisation Com2 frac dimin
1038	Utilisation Nav1 entier augm
1039	Utilisation Nav1 entier dimin
1040	Utilisation Nav1 frac augm
1041	Utilisation Nav1 frac dimin
1042	Utilisation Nav2 entier augm
1043	Utilisation Nav2 entier dimin
1044	Utilisation Nav2 frac augm
1045	Utilisation Nav2 frac dimin
1046	Utilisation Adf1 entier augm
1047	Utilisation Adf1 entier dimin
1048	Utilisation Adf1 frac augm
1049	Utilisation Adf1 frac dimin
1050	Utilisation Adf2 entier augm
1051	Utilisation Adf2 entier dimin
1052	Utilisation Adf2 frac Augm
1053	Utilisation Adf2 frac dimin
1054	Xpndr low NN dimin
1055	Xpndr low NN augm
1056	Xpndr high NN dimin
1057	Xpndr high NN augm
1058	Freeze pos on
1059	Freeze pos off

1060	Freeze pos toggle
1061	Engine 1 Autostart
1062	Engine 2 Autostart
1063	Engine 3 Autostart
1064	Engine 4 Autostart
1065	Throttles off
1066	Throttles on
1067	Throttles toggle
1068	PVT voice transmit on (pour Squawbox 3.0.4 ou ultérieur)
1069	PVT voice transmit off (pour Squawbox 3.0.4 ou ultérieur)
1070	Appui touche et Relâchement (param = keycode+256*Shift code , ou JsBk)
1071	Appui touche et Relâchement (param = keycode+256*Shift code , ou JsBk)
1072	Relâchement touche (param = keycode+256*Shift code , ou JsBk)
1073	FSUIPC4 affiche basculement fenêtre (window toggle)
1074	Réglage densité trafic aérien
1075	Réglage densité trafic GA
1076	Réglage densité trafic Shipping
1077	Réglage densité couche nuageuse
1078	Réglage nuages simple/complexe
1079	Zappeur Trafic
1080	Bascule (toggle) Wheel trim (pour ajuster la souris)
1081	Wheel trim plus vite
1082	Wheel trim moins vite
1083	Bascule (toggle) Wheel speed trim
1084	Lua Kill All
1085	Traffic Zapall
1086	FollowMe please (c’est-à-dire demande (request)) (nécessite FollowMe2)
1087	Efface FollowMe (nécessite FollowMe2)
1088	Continue FollowMe (nécessite FollowMe2)
1089	Bascule (toggle) AutoDeleteAI
1090	AutoDeleteAI on
1091	AutoDeleteAI off
1092	Re-SimConnect (réinitialise l’interface SimConnect)
1093	Gamme Efis ND augm (par défaut B738 et A321)
1094	Gamme Efis ND dimin (par défaut B738 et A321)
1095	Mode Efis ND augm (par défaut B738 et A321)
1096	Mode Efis ND dimin (par défaut B738 et A321)
1097	Map item Efis ND augm (par défaut B738 et A321)
1098	Map Item Efis ND dimin (par défaut B738 et A321

1099	VORADF1 Efis augm (par défaut B738 et A321)
1100	VORADF1 Efis dimin (par défaut B738 et A321)
1101	VORADF2 Efis augm (par défaut B738 et A321)
1102	VORADF2 Efis dimin (par défaut B738 et A321
1103	Efis 738 ND centre (par défaut B738)
1104	Efis 738 ND arc (par défaut B738)
1105	Bascule (toggle) Efis A321 InHg/hPA (par défaut A321)
1106	Bascule (toggle) mode ILS Efis A321 (par défaut A321)
1107	Bascule (toggle) changement de taux (rate) alt AP (par défaut A321)
1108	Réglage gamme Efis ND (paramètre 0-7 pour 738 , 0-5 pour A321) (par défaut B738 et A321)
1109	Réglage mode Efis ND (paramètre 0-2 pour 738 , 0-3 pour A321) (par défaut B738 et A321)
1110	Réglage map item Efis ND (paramètre 0-3) (par défaut B738 et A321)
1111	Réglage VORADF1 Efis (paramètre 0-2) (par défaut B738 et A321)
1112	Réglage VOADF2 Efis (paramètre 0-2) (par défaut B738 et A321)
1113	Réglage Efis A321 InHg/hPA (paramètre 0-1) (A321)
1114	Liste des variables en log (« L :vars ») du tableau de bord local lors d’un changement d’avion
1115	IYP Listen On
1116	IYP Listen Off
1117	IYP COmeFly Actif
1118	IYP Come Fly Inactif
1930	FSUIPC bank hold off
1931	FSUIPC bank hold on
1932	FSUIPC bank hold set
1933	FSUIPC bank hold toggle
1934	FSUIPC mach hold off
1935	FSUIPC mach hold on
1936	FSUIPC mach hold set
1937	FSUIPC mach hold toggle
1938	FSUIPC pitch hold off
1939	FSUIPC pitch hold on
1940	FSUIPC pitch hold set
1941	FSUIPC pitch hold toggle
1942	FSUIPC speed hold off
1943	FSUIPC speed hold on
1944	FSUIPC speed hold set
1945	FSUIPC speed hold toggle

2010 PM MCP SPD push on B747
2011 PM MCP HDG sel on B747
2012 PM MCP ALT push on B747
2013 –
2014 –
2015 –
2016 –
2017 PM MCP FD2 off
2018 PM MCP FD2 on
2019 PM MCP A/T on
2020 PM MCP A:T off
2021 PM MCP THR mode button
2022 PM MCP SPD mode button
2023 PM MCP Mach/IAS sel
2024 PM MCP FLCH mode button
2025 PM MCP A:T off
2026 PM MCP THR mode button
2027 PM MCP HDG mode button
2028 PM MCP LOC mode button
2029 PM MCP APP mode button
2030 PM MCP ALT mode button
2031 PM MCP VS mode button
2032 PM MCP AP1 (L) button
2033 PM MCP AP2 © button
2034 -
2035 -
2036 PM MCP AP3 ® button
2037 PM MCP FD1 off
2038 PM MCP FD1 on
2039 -
2040 PM MCP AP Disc (pas 747)
2041 PM MCP AP Eng (pas 747)
2042 PM MCP AP Disc (747 seulement)
2043 -
2044 -
2045 -
2046 -
2047 -
2048 -
2049 PM AB LS button
2050 PM AB STD QNH rel (push)
2051 PM AB STD QNH set (pull)
2052 PM AB SPD button push

2053	PM AB SPD button pull
2054	PM AB HDG button push
2055	PM AB HDG button pull
2056	PM AB ALT button push
2057	PM AB ALT button pull
2058	PM AB VS button push
2059	PM AB VS button pull
2060	PM AB EXPED button
2061	PM AB TRKFPA button
2062	–
2063	–
2064	PM PFD Decision Ht Dimin
2065	PM PFD Decision Ht Augm
2066	PM MCP Hdg Dimin 1
2067	PM MCP Hdg Augm 1
2068	PM MCP Hdg Dimin 10
2069	PM MCP Hdg Augm 10
2070	PM MCP Alt Dimin 100
2071	PM MCP Alt Augm 100
2072	PM MCP Alt Dimin 1000
2073	PM MCP Alt Augm 1000
2074	PM MCP Spd Dimin 1/.01
2075	PM MCP Spd Augm 1/.01
2076	PM MCP Spd Dimin 10/.10
2077	PM MCP Spd Augm 10/.10
2078	PM MCP V/S Dimin 100
2079	PM MCP V/S Augm 100
2080	PM MCP Crs Dimin 1
2081	PM MCP Crs Augm 1
2082	PM QNH Dimin 0.01/1
2083	PM QNH Augm 0.01/1
2084	PM ND Range Dimin
2085	PM ND Range Augm
2086	PM ND Mode Dimin
2087	PM ND Mode Augm
2088	PM ND2 Range Dimin
2089	PM ND2 Range Augm
2090	PM ND2 Mode Dimin
2091	PM ND2 Mode Augm
2092	-
2093	-
2094	-
2095	-

2096 PM AB ND ILS Mode
2097 PM ND Map Arc Mode
2098 PM ND Map Ctr Mode
2099 PM ND Rose Mode
2100 PM ND Map Plan Mode
2101 PM ND Range 10
2102 PM ND Range 20
2103 PM ND Range 40
2104 PM ND Range 80
2105 PM ND Range 160
2106 PM ND Range 320
2107 PM ND Range 640
2108 PM ND VOR Display
2109 PM ND NDB Display
2110 PM ND WPT Display
2111 PM ND ARPT Display
2112 PM ND DATA Display
2113 PM ND POS Display
2114 PM AB ND VOR1 on
2115 PM AB ND ADF1 on
2116 PM AB ND VORADF1 off
2117 PM AB ND VOR2 on
2118 PM AB ND ADF2 on
2119 PM AB ND VORADF2 off
2120 PM AB ND Metric
2121 PM AB ND HDGVS/TRKFPA
2122 PM AB ND THR TOGA
2123 PM AB ND THR FLX/MCT
2124 PM AB ND THR CLB
2125 PM AB ND THR IDLE
2126 PM AB ND THR REV IDLE
2127 PM AB ND THR MAX REV
2128 PM AB ND2 ILS Mode
2129 PM ND2 Map Arc Mode
2130 PM ND2 Map Ctr Mode
2131 PM ND2 Rose Mode
2132 PM ND2 Map Plan Mode
2133 PM ND2 Range 10
2134 PM ND2 Range 20
2135 PM ND2 Range 40
2136 PM ND2 Range 80
2137 PM ND2 Range 160
2138 PM ND2 Range 320

2139 PM ND2 Range 640
2140 PM ND2 VOR Display
2141 PM ND2 NDB Display
2142 PM ND2 WPT Display
2143 PM ND2 ARPT Display
2144 PM ND2 DATA Display
2145 PM ND2 POS Display
2146 PM AB ND2 VOR1 on
2147 PM AB ND2 ADF1 on
2148 PM AB ND2 VORADF1 off
2149 PM AB ND2 VOR2 on
2150 PM AB ND2 ADF2 on
2151 PM AB ND2 VORADF2 off
2152 PM AB ND2 Metric
2153 PM AB ND2 HDGVS/TRKFPA
2154 –
2155 –
2156 –
2157 –
2158 –
2159 –
2160 PM EICAS Show Controls
2161 PM EICAS Standby Gauge
2162 PM EICAS Page Dimin
2163 PM EICAS Page Augm
2164 PM EICAS Synoptic Dimin
2165 PM EICAS Synoptic Augm
2166 PM AB ND ILS Mode
2167 PM ND Plan Wpt Dimin
2168 PM ND Plan Wpt Augm
2950 Bascule (toggle) PM Elec All
2951 Bascule (toggle) PM Elec PFD
2952 Bascule (toggle) PM Elec ND
2953 Bascule (toggle) PM Elec EICAS
2955 Bascule (toggle) PM Elec PFD2
2956 Bascule (toggle) PM Elec ND2
2957 Bascule (toggle) PM Elec Stdby
2966 PM Elec All ON
2967 PM Elec PFD ON
2968 PM Elec ND ON
2969 PM Elec EICAS ON
2971 PM Elec PFD2 ON
2972 PM Elec ND2 ON

2974 PM Elec Stdby ON
2982 PM Elec All OFF
2983 PM Elec PFD OFF
2984 PM Elec ND OFF
2985 PM Elec EICAS OFF
2987 PM Elec PFD2 OFF
2988 PM Elec ND2 OFF
2990 PM Elec Stdby OFF
2994 PM Whazzup keys (par Param) , voir liste PM Offsets , 542E
2995 PM Quickmap keys (par Param) , voir Liste PM Offsets , 542C
2996 PM GC keys (par Param) , voir Liste PM Offsets , 542A
2997 PM CDU keys (par Param) , voir Liste PM Offsets , 5428

Note : toutes les entrées « keys » vers les modules PM fournissent des chemins efficaces à partir d'appuis touches clavier vers eux , où qu'ils puissent être sur le réseau . Chez eux , le paramètre est le code appui touche (keystroke) (voir la liste précédemment dans ce document) , plus des valeurs spécifiques PM-définies pour les shifts , donc :
256 pour Shift , 512 pour Ctrl , 1024 pour Alt .
Vous n'avez pas à vous soucier de changer d'autres bits lorsque 2 codes sont les mêmes – FSUIPC s'en occupe automatiquement .

2998 PM MCP Kcodes (par Param) , voir la liste d'Offsets Pm , 04F2

Cette façon de contrôler le MCP de PM peut offrir des fonctionnalités qu'on ne trouve pas partout . Le paramètre est le numéro (number) utilisé dans les codes Elan Informatique « Knnn »envoyés normalement au MCP via une connexion port série (serial connection) . Voici la liste de ceux qui sont connus actuellement , mais référez-vous je vous prie au document PM Offsets pour une mise-à-jour .

10 SPDP (SPD pushbutton 747 MCP , Speed Intervention sur B737 MCP)
11 HDGP (heading SEL pushbutton 747 MCP , utiliser 25 pour HDG HOLD , utiliser 25 pour HDG SEL sur le 737)
12 ALTP (ALT pushbutton 747 MCP , Altitude Intervention sur 737)
17 FDON (switch on sur FD de first Officer)

18 FDFF (switch off sur FD de first Officer)
19 ATON (swich on)
20 ATFF (swich off)
21 THR
22 SPD
23 MACH
24 FLCH
25 HDG K025
26 VNAV K026
27 LNAV K027
28 LOC K028
29 APP K029
30 ALT K030
31 VS K031
32 AP1 K032
33 AP2 K033
34 CWSA K034
35 CWSB K035
36 AP3 K036
37 FDON K037 (swich on FD du Commandant de bord)
38 FDFF K038 (swich off FD du Commandant de bord)
40 APDI (désengage AP – pas utilisé 747-400 MCP)
41 APEN (engage AP – pas utilisé 747-400 MCP)
44 FPV
45 MTRS
46 CTR ND
47 TFC (TCAS)
48 RST
49 STD
50 VOR1
51 1ADF1
52 OFF1
53 VOR2
54 ADF2
55 OFF2
62 IN
63 HPA
64 setDH
65 setMDA
66 APP ND
67 VOR ND
68 MAP ND
69 PLN ND

70 VOR
72 OFF1
73 VOR2
74 ADF2
75 OFF2
80 STA
81 WXR
99 DISC (déconnecte 747)
144 FPV Copilote
145 MTRS Copilote
147 TFC (TCAS Copilote)
148 RST (RST Copilote)
149 STD (Copilote STD)
170 VOR1 F/O
171 ADF1 F/O
172 OFF1 F/O
173 VOR2 F/O
174 ADF2 F/O
175 OFF2 F/O

2999 Commandes GC Projet Magenta . Param spécifie l'action (comme montré ci-dessous , liste provenant de la publication de Project Magenta Offsets , avec autorisation) [*ajoutez 100 pour Premier Officier GC , sinon ce sera Commandant de Bord*]

Airbus
1 MAP (Captain side , 101 F/O side)
2 NAV (Captain side , 102 F/O side)
3 VOR (Captain side , 103 F/O side)
4 PLAN (Captain side , 104 F/O side)
5 ILS Mode

Boeing "classic Modes"
1 MAP ARC
2 MAP CTR
3 VOR
4 MAP PLAN

Nouveaux Modes ND (!)
1 MAP
3 VOR
4 PLN

5 APP
6 CTR Pushbutton
7 Force display to 8 Modes (APP/VOR/MAP/PLN)

8 Montre commandes dans EICAS/ECAM
9 Cache commandes dans EICAS/ECAM
10 PFD/ND →PFD→ND (comme F4,F1,F2 dans GC)
11 PFD/EICAS
12 EICAS avec Standby
13 EICAS sans Standby
14 FPV (Boeing)
15 EICAS/ND
19 Commandes Toggle dans EICAS/ECAM
20 Page Augm Moteurs
21 Page Dimin Moteurs
22 Toggle no smoking
23 Toggle Seatbelts
24 Toggle Page Vue d'ensemble (overview)
25 Toggle affichage RMI/HSI dans ND MAP ARC type-boeing
26 Toggle Metric
28/29 ND Mode Augm/Dimin pour Airbus
30 Moteurs Page 0 (primaire)
31 Moteurs Page 1
32 Moteurs Page 2
..
39 Moteurs Page 9 (si définie)
40 Portée (range) 5 NM
41 Portée 10 NM
42 Portée 20 NM
43 Portée 40 NM
44 Portée 80 NM
45 Portée 160 NM
46 Portée 320 NM
47 Portée 640 NM
48 Portée Dimin
49 Portée Augm
50 TCAS Off
51 TCAS Alt
52 TCAS Appel (Callsign)
53 TCAS All
54 Toggle TCAS Off/Alt
55 Affiche valeurs MCP dans EICAS (Boeing)
56 cache valeurs MCP dans EICAS (Boeing)

57 Mode PLAN après waypoint
58 Mode PLAN avant waypoint
60 Affiche Page Vue D'ensemble dans ND
61 Cache Page Vue d'ensemble dans ND
62 Chronomètre (timer) Set/reset (AB Glass Cockpit)
70 Affiche WXR
71 cache WXR
72 Toggle WXR
73 VORADFL OFF
74 ADFL ON
75 VORL ON
76 VORADFR OFF

77 ADFR ON
78 VORR ON
80 Affichage Terrain On
81 Affichage Terrain Off
82 Toggle Affichage Terrain
83 Change type Terrain
84 Change Mode/couleur Terrain
85 Change Taille Terrain
86 Terrain 3D
90 STA
91 VOR
92 NDB
93 WPT
94 ARPT
95 DATA
96 POS

321 Affichage Page Synoptic/system Dimin
322 Affichage Page Synoptic/system Augm

(Boeing)
Secondairement Pages EICAS et fonctions 747
(777)
301 ENG
302 STAT
303 ELEC
304 FUEL (*777 :HYD*)
305 ECS (*777 :FUEL*)
306 HYD (*777 :AIR*)
307 DRS (*777 :DOORS*)

308 GEAR
309 ---- (*777:FCTL*)
310 CANC
311 RCL

(Boeing)
401 Caution On (voir 0x4FE)
402 Caution Reset

411 Toggle Affichage FuelUsed
412 ShowFuelUsed on
413 ShowFuelUsed off
414 Reset FuelUsed = 0

(les deux)
421 Toggle no smoking
422 No smoking On
423 No smoking Off
424 Toggle Seatbelts
425 Seatbelts On
426 Seatbelts Off

(Airbus)
Secondairement Pages EICAS et fonctions AB
301 ENG
302 BLEED
303 PRESS
304 ELEC
305 HYD
306 FUEL
307 APU
308 COND
310 DOOR
311 WHEEL
312 F/CTL
313 ALL
314 CLR
315 STS
316 RCL
317 CLR
318 EL/DC (A330/340)
319 C/B (A330/340)

X0100zzzz Offset Byte Set (offset= zzzz), hexadecimal
X0200zzzz Offset Word Set (offset= zzzz), hexadecimal
X0300zzzz Offset Dword Set (offset= zzzz), hexadecimal
X0500zzzz Offset Byte Setbits (offset= zzzz), hexadecimal
X0600zzzz Offset Word Setbits (offset= zzzz), hexadecimal
X0700zzzz Offset Dword Setbits (offset= zzzz), hexadecimal
X0900zzzz Offset Byte Clrbits (offset= zzzz), hexadecimal
X0A00zzzz Offset Word Clrbits (offset= zzzz), hexadecimal
X0B00zzzz Offset Dword Clrbits (offset= zzzz), hexadecimal
X0D00zzzz Offset Byte Togglebits (offset= zzzz), hexadecimal
X0E00zzzz Offset Word Togglebits (offset= zzzz), hexadecimal
X0F00zzzz Offset Dword Togglebits (offset= zzzz), hexadecimal
X1100zzzz Offset UByte Increment (offset= zzzz), hexadecimal*
X1200zzzz Offset UWord Increment (offset= zzzz), hexadecimal*
X2100zzzz Offset UByte Decrement (offset= zzzz), hexadecimal*
X2200zzzz Offset UWord Decrement (offset= zzzz), hexadecimal*
X3100zzzz Offset SByte Increment (offset= zzzz), hexadecimal*
X3200zzzz Offset Sword Increment (offset= zzzz), hexadecimal*
X4100zzzz Offset SByte Decrement (offset= zzzz), hexadecimal*
X4200zzzz Offset Sword Decrement (offset= zzzz), hexadecimal*
X5100zzzz Offset Byte Cyclic Increment(offset= zzzz), hexadecimal*
X5200zzzz Offset Word Cyclic Increment (offset= zzzz), hexadecimal*
X6100zzzz Offset Byte Cyclic Decrement (offset= zzzz), hexadecimal*
X6200zzzz Offset Word Cyclic Decrement (offset= zzzz), hexadecimal*
X7000zzzz Offset Float32 Set/1000 (offset= zzzz), le paramètre est divisé par 1000
X7400zzzz Offset Float64 Set/1000 (offset= zzzz), le paramètre est divisé par 1000
X7800zzzz Offset Float32 Inc/1000 (offset= zzzz), le paramètre est divisé par 1000
X7C00zzzz Offset Float64 Inc/1000 (offset= zzzz), le paramètre est divisé par 1000
(pour Diminuer , utiliser un paramètre négati dans la commande Increment)

* les valeurs augm/dimin avec un .nnn fonctionnent sur des valeurs non signées (U) ou signées (S) , et ont un paramètre avec la limite signée

ou non signée dans les 16 bits supérieurs (upper) et la quantité (amount) increment/decrement (toujours non signés) dans les 16 bits inférieurs (low)

COMMANDES MACRO

FSUIPC4 lira tout fichier de type .mcro dans le répertoire Modules . De tels fichiers contiennent des définitions de commandes supplémentaires, listées et assignables dans les boites de dialogues d'assignement des Keys , Buttons et Axis de FSUIPC4 . Tous les fichiers macro sont également relus et réinstallés en appuyant sur le bouton « reload » de n'importe quelle de ces 3 boites de dialogue .

Il est important que le nom de fichier (xxxx.mcro) soit unique dans les 16 premiers caractères , puisque ce sera utilisé comme partie du nom des commandes supplémentaires dans les menus déroulants . Le mieux est de conserver des noms courts – probablement le nom du programme ou de la fonction dans le programme pour lesquelles les commandes ont été ajoutées .

A l'intérieur d'un fichier macro , il pourrait juste y avoir une section appelée [Macros] , qui devrait contenir des définitions de commandes numérotées, avec des noms également jusqu'à 16 caractères . Ces noms doivent seulement être propres à ce fichier .

Voici un exemple , qui concerne un éventuel swich Mode ND glass cockpit Project Magenta :

```
[Macros]
1=MAP Capt=C2999,1
2=NAV Capt=C2999,2
3=VOR Capt=C2999,3
4=PLN Capt=C2999,4
5=APP Capt=C2999,5
6=CTR Capt=C2999,6
101=MAP F/O=C2999,101
102=NAV F/O=C2999,102
103=VOR F/O=C2999,103
104=PLN F/O=C2999,104
105=APP F/O=C2999,105
106=CTR F/O=C2999,106
```

Notez que les numéros à gauche peuvent ne pas être contigus , mais doivent être dans la gamme 1-999 inclus. Ils seront utilisés de façon

interne , et dans le fichier FSUIPC4.INI , pour identifier la commande à l'intérieur du fichier .

Supposons que l'exemple ci-dessus survienne dans un fichier nommé « PM GC.mcro » . Les noms qui pourraient apparaître par ordre alphabétique dans les menus déroulants de FSUIPC4 pourraient être :

PM GC :APP Capt
PM GC :APP F/O
PM GC :CTR Capt
PM GC :CTR F/O
PM GC :MAP Capt
PM GC :MAP F/O
PM GC :PLN Capt
PM GC :PLN F/O
PM GC :VOR Capt
PM GC :VOR F/O

La valeur assignée à chaque commande est soit une autre commande (toute commande supplémentaire FS ou FSUIPC incluant des commandes offset et même des commandes macro – voir plus loin) , ou un appui touche clavier , c'est-à-dire :

Soit : Cn,p (numéro de commande , paramètre , optionnellement en hexadécimal si précédé d'un x

Ou bien : Kk,s (code touche et shifts).

Les deux sont exactement tels qu'ils ont déjà été définis dans les commandes Buttons – voir la section précédente sur la programmation de Bouton .

Références de commandes macro

Les commandes macro sont représentées de façon interne de la même façon que les commandes offsets de FSUIPC , en utilisant des bits valeur-haute dans le numéro de commande . Cependant , la représentation dans les fichiers Macro et dans le fichier .INI est la suivante :

Mm :n

Où m est le numéro du fichier Macro (voir ci-dessous) et n est le numéro de commande depuis le fichier , comme décrit ci-dessus .

Les numéros de Fichier Macro sont assignés par FSUIPC lorsqu'il charge le fichier en mémoire . Ceux-ci sont rappelés en mémoire (remembered) dans le fichier .INI dans une nouvelle section [Macrofiles]. Par exemple , dans le cas ci-dessus , vous pouvez obtenir :

[Macrofiles]
1=PM GC

Créant « PM GC.mcro » fichier numéro 1 pour tous objectifs (purpose) référence

Il est important de noter que différents utilisateurs sélectionneront différents fichiers macros dans des ordres différents . S'ils souhaitent échanger des assignements de Bouton , ils auront besoin de réassigner toutes les commandes macro après avoir créé les mêmes sections [Macrofiles] ou tout du moins les mêmes pour les fichiers qu'ils ont en commun .

Actions multiples dans une seule commande macro

Une commande Macro n'est pas limitée à une seule action en résultant . Si l'on demande plus d'une action , alors plusieurs lignes sont utilisées dans la définition , comme suit :

n=<name>
n.1=action1
n.2=action2
etc.

Par exemple , considérons un fichier "Menu.mcro" contenant ces définitions :

[Macros]
1=Display
1.1=K79,16 ;O
1.2=K69,8 ;E
1.3=K68,8 ;D
2=FSUIPC

2.1=K68,16 ;Alt D
2.2=K70,8 ;F

Ceci ajoute 2 commandes , “ Menu: Display” et “Menu: FSUIPC” . la première utilise ALT + O , E , D appuis touches clavier pour appeler la boite de dialogue des réglages de l’Affichage FS , la seconde utilise Alt + D , F pour appeler les options FSUIPC4 .

Notez qu’il existe au total une limite de 2000 paramètres numérotés dans le fichier macro – donc , par exemple , 999 numéros macro (1-999 , le maximum) avec une moyenne de 2 actions par numéro flirteraient un peu trop avec la limite . Quel que soit le cas , de grands fichiers ne sont pas bons puisque la liste de menus déroulants sera pleine de commandes supplémentaires qui commenceront toutes par le même nom de fichier . Le mieux est de les fractionner en groupes fonctionnels avec des noms significatifs pour rendre les commandes plus faciles à localiser .

Paramètre laisser-passer (passing)

Normalement , et d’évidence dans tous les exemples ci-dessus , n’importe quel paramètre réglé pour une commande Macro , lorsqu’il est assigné dans la boite de dialogue Buttons & Keys , pourrait être abandonné comme non pertinent . Cependant il existe une fonctionnalité pour lui permettre d’être utilisé .

Si la partie paramètre de n’importe laquelle des commandes définie dans la macro est manquante (omitted) la valeur paramètre de la macro qui appelle est substituée .

Un exemple un peu stupide : si vous voulez une commande générale PM GC mais pas une qui a déjà été nommée dans FSUIPC , vous pourriez la définir comme :

7=by param=C2999

Ce qui pourrait apparaître dans les menus déroulants en tant que « PM GC : by param » , et le paramètre assigné par l’utilisateur pourrait être utilisé dans l’opération C2999 . Notez que dans les définitions lignes-multiples , la même valeur paramètre se substitue à chaque valeur paramètre manquante (omitted) .

Une conséquence intéressante , c'est la possibilité de définir des commandes d'axes . Un autre exemple stupide : si 1 définit une macro comme ceci :

8=Flaps=C66534 ;la commande FS 66534 le réglage d'axe des volets

Et puis l'assigner à un Axe dans le menu déroulant des Assignements d'axes , l'axe que j'aurai assigné opèrera exactement comme l'axe Réglage d'Axe des Volets .

Ça ne vous semblera pas si futile lorsque vous réaliserez que vous pourrez avoir des mélanges en lignes-multiples de commandes et d'appuis touches clavier également produites par la même Macro . Je suis sûr qu'il y aurait pléthore d'idées pour utiliser cette « fonctionnalité » (qui s'est actuellement fâchée avec l'implémentation par accident davantage que par dessein !)

Macros souris

Une autre fonctionnalité des fichiers Macro , c'est leur aptitude à ajouter des commandes à votre arsenal fait d'interrupteurs , de boutons , tableaux de bords de FS , et de jauges (pour la plupart des add-ons) , qui ne peuvent être utilisés qu'avec la souris .En outre , cette fonctionnalité peut actuellement être utilisée même sans recourir à préparer manuellement directement les fichiers Macros – cette partie est semi-automatisée via les boutons Mouse Macro dans les onglets de l'option Buttons & Keys de FSUIPC4 . Des détails sur les fonctionnalités automatisées sont fournis dans le Guide de l'Utilisateur . Nous nous concentrons ici juste sur le fichier lui-même , le format des lignes mouse macro .

Notez qu'un simple fichier macro peut contenir n'importe quel mélange de souris et d'autres macros . En fait les actions de la souris , des commandes (control) et des appuis touches clavier peuvent être mélangées (mixed) et combinées en une simple macro . Bien sûr , ceci n'arrive pas aux macros générées automatiquement .

Cette fonctionnalité souris ajoute le format pour le moins obscur :

R<module> :<rect#>,<mousefalg>

A ceux déjà décrits pour les Touches clavier (K) les commandes (C) , et en avant (onward) les références Macro (M) . Ici le « R » est pour le **R**ectangle de la souris , parce que c'est via des rectangles spécifiques sur l'écran que FS reconnaît les demandes de la souris . Un « M » pour mouse aurait été mieux , mais il avait déjà été utilisé pour Macro .

Je vais expliquer maintenant ce que ces valeurs , dans cette spécification , veulent dire actuellement , mais en général , aucun utilisateur ne sera concerné puisqu'elles doivent toutes être remplacées par le concepteur de jauges (le fournisseur d'add-ons tableaux de bords) ou bien , de manière plus habituelle , être générées automatiquement pour vous par FSUIPC, à travers l'utilisation de la souris en mode création macro mouse .

Donc , pour ce qui concerne la spécification de l'action de la souris :

<**module**> est optionnel . C'est une référence au nom de fichier Gauge ou DLL , c'est-à-dire à la partie du tableau de bord à laquelle s'appliquera l'action de la souris . Il s'agit d'une référence numérique à une autre ligne qui doit aussi être présente quelque part dans la sections [Macros] du fichier .MCRO , quelque chose comme :

ModuleN = « nom de Jauge ou DLL »

Où « N » va de 1 à 99, ou bien peut être manuqant (auquel cas ça donne : « Module= » … » »)

Si la partie <module> de l'action de la souris est manquante , le Module référencé sera celui sans aucun numéro . En outre , c'est tout simplement N : , référent directement au module .

La partie <**rect#**> est la seule partie obligatoire . Il s'agit soit d'une référence au numéro « MouseRect » dans les tables du Module – tel que « n » référant au énième rectangle , à compter de 0 – ou une référence directe à la Fonction Souris dans le Module , tel que « Xxxxx*Xxxxx », où les parties « xxxx » réfèrent respectivement : à un offset hexadécimal et à un check-word L'offset vient d'une adresse chargée en mémoire depuis le Module , et le check-word , ce sont les 16 bits autour du point d'entrée de la fonction souris : 8 bits avant et 8 bits après . Le check-word est une mesure de sécurité , au cas où la macro serait utilisée avec une version différente de la même jauge ou de la même DLL .

Finalement , la partie <**mouseflag**> fournit l'action de la souris actuellement requise pour faire fonctionner la fonctionnalité . Elle est encodée comme un numéro , parfois un de ceux-ci :

31 MOUSE_RIGHT SINGLE
30 MOUSE_ MIDDLE SINGLE
29 MOUSE_ LEFT SINGLE
28 MOUSE_ RIGHT DOUBLE
27 MOUSE_ MIDDLEDOUBLE
26 MOUSE_ LEFT DOUBLE
21 MOUSE_ DOWN_ REPEAT
19 MOUSE_ RIGHTRELEASE*
18 MOUSE_ MIDDLERELEASE*
17 MOUSE_ LEFTRELEASE*
14 MOUSE_ WHEEL_UP
13 MOUSE_ WHEEL_DOWN
11 MOUSE RIGHT SINGLE
31 MOUSE_LEAVE*

Parmi ceux ci , 29 est de loin le plus commun et est adopté (assumed) quand le paramètre est manquant .Les valeurs actuellement assimilées aux flags de souris par ces noms se trouvent dans FS Gauge C/C++ SDK .

Les flags marqués d'une astérisque (*) ne peuvent pas être générés automatiquement par FSUIPC puisqu'ils réfèrent aux boutons de souris relâchés .Cependant ils peuvent s'avérer nécessaires pour quelques implémentations de switchs , et vous pourriez avoir besoin de les ajouter par vous-même – il y a des exemples dans le Guide de l'Utilisateur .

Pour remettre les choses dans leur contexte , voici quelques exemples actuels . le premier concerne le panneau de plafond (overhead) du 737 NG de PMDG pour FS9 :

Module= "PMDG_737NG_Overhead.gau »
1=Batt=RX3170*X8b90
Module1="PMDG_737NG_OHD.APU.GAU"
40=APU=R1:1

Si ces lignes sont dans un fichier .MCRO chargé en mémoire , qui s'appelle « 737 OHD » , alors les commandes par menus déroulants Buttons & Keys listeront «737 OHD :Batt » , qui fera fonctionner le switch de la batterie, et « 737 OHD : APU » qui fera fonctionner le switch APU .

Ils feront quelque chose seulement si la Jauge Panneau de plafond (overhead gauge) est chargée en mémoire – c'est-à-dire si l'avion est utilisé . Remarquez que la jauge panneau de plafond elle-même n'a pas à être visible .

Voici un extrait du fichier Macro pour l'add-on jauge/DLL « APchart » :

Window= »Airport Chart »
Module= »APchart.dll »
1=Show/Hide=C66506,10000
…
7=Knob 1 Down=R20,14
8=Knob 1 Up=R20,13

Ce fichier a une commande non-souris inclue pour afficher et cacher la fenêtre AP chart . ça utilise la commande « PANEL ID SET » avec le numéro PANEL ID 10000 comme paramètre (glané à partir du fichier Panel.cfg) . Il a également un couple d'entrées affichées qui sont commandées par la molette de la souris (mouse wheel) .

Mais notez ce nouveau paramètre :

Window= »window title »

Il a besoin d'être présent alors qu'il permet seulement de comprendre les commandes avec la fenêtre à la fois ouverte et visible . ça s'applique à APchart où zoomer et se déplacer sur la carte serait idiot si on ne voyait pas ce que l'on fait . Vous trouverez des noms de Fenêtres pour des parties de tableaux de bord dans le fichier Panel.CFG . La macro automatique de souris qui génère des fonctionnalités dans FSUIPC n'ajoute jamais un paramètre Window, donc ça peut être une bonne raison pour que vous éditiez un fichier .MCRO .

Accès aux variables locales des Jauges (L :vars) par MACRO

Les variables locales nommés (« L :<name> ») , auxquelles je me réfèrerai en tant que « **Lvars** » peuvent maintenant être listées dans le Log , écrite via des Macros , et manipulées à la fois avec les lectures et les écritures à travers des extensions vers la Librairie ipc Lua .

Le listing log est obtenu à partir des tableaux de bords actuellement chargés en mémoire par une commande nouvellement assignée dans les menus déroulants , appelée « **List local panel variables** » .

Elle demande à FSUIPC4 de lister tous les Lvars trouvés dans le fichier log FSUIPC4 , par L:name et par valeur courante. Par exemple , voici le listing Log pour le Cessna 172 par défaut , avec le G1000 :

Aircraft= »Cessna Skyhawk 172SP G1000 »
Le tableau de bord inclut ces variables locales:
L :CDI Source Selected = 1.000000
L :time hdg bug changed = 0.000000
L:time crs changed = 0.000000
L:SelectedNav2 = 0.000000
L : SelectedCom2 = 0.000000
L :pfd HDGKnob pressed = 0.000000
L:COM1 Mic pressed = 0.000000
L:COM2 Mic pressed = 0.000000
L :NAV1 pressed = 0.000000
L :NAV2 pressed = 0.000000
L:MKR pressed = 0.000000
L:DME pressed = 0.000000
L :COM1 pressed = 1.000000
L :COM2 pressed = 0.000000
L:Reversionary pressed = 0.000000
L:map_Zoomstep = 0.000000
L :map_ZoomFactor = 0.000000
L :MapInit = 0.000000
L:LayerAirports = 0.000000
L:LayerAirspaces = 0.000000
L :LayerTerrain = 0.000000
L :LayerVORs = 0.000000
L:LayerILSs = 0.000000
L:LayerNDBs = 0.000000
L :LayerLowAirways = 0.000000
L :LayerTags = 0.000000
L:LayerCompass = 0.000000
L:LayerIntersections = 0.000000
L :LayerRangeRings = 0.000000
L :VehicleObjectDetail = 0.000000
L:Filter = 0.000000
L:LastLandingLightPosition = 0.000000
L:EmergencyThrottleInUse = 0.000000

L:Engine1ThrottlePosition = 24.404907

Notez que toute ce que faire FSUIPC , c'est lister ce qu'il trouve . On peut s'interroger si les valeurs servent à quelque chose ou pas – elles appartiennent à la jauge et la façon dont elles sont utilisées , manipulées , etc. variera énormément . Essayez des choses , si vous le souhaitez , par tous les moyens , mais ne pensez pas que la solution est là , qui vous attend …

Quelques unes des **Lvars** réellement utiles , spécialement celles pour les tableaux de bord EFIS du B738 par défaut et de l'A321 ont donné lieu à des commandes spécifiques FSUIPC4 et aussi des offsets , de telle sorte qu'ils peuvent être manipulés directement par des programmes d'application . Mais pour les autres , vous aurez besoin d'utiliser des fichiers Macro ou des plugs-in Lua , comme décrits ci après .

Macros pour changer les Lvars

La fonctionnalité Macro pour faire fonctionner des **Lvars** peut seulement être utilisée en éditant des fichiers macro et en les construisant manuellement . Le format est :

N=L :name=ACTION

Où ACTION peut être **Set** , ou **Inc** , ou **Dec** ou **Cyclic** ou **Toggle** (mais on n'a besoin que des 3 premières lettres) .

Set copie le paramètre dans l'invocation Macro vers le Lvar identifié . Alternativement , une valeur peut être donnée ici , par « **Set,n** » . Les valeurs sont limitées à : - 32768 à 32767 .

Inc augmente la valeur , ici le paramètre (explicite ou fourni) donne la limite supérieure , qui peut être égalée , mais non dépassée

Dec définit la valeur , avec un paramètre qui règle la limite inférieure

Cyclic le même que **Inc** , mais une fois que la limite est atteinte , la valeur suivante est 0

Toggle change la valeur en 0 si c'est non zéro , ou en 1 si c'est 0

Le format multi-lignes peut encore être utilisé avec les macros Lvars , comme suit :

N=L :name
N.1=action1
N.2=action2
… etc .

Accès Lua à Lvars

Les fonctionnalisé Lua sont **: ipc.readLvar , ipc.writeLvar , ipc.getLvarName , et ipc.getLvarId** .Elles sont toutes décrites dans la documentation de la Librairie Lua , et un échantillon plug-in Lua est fourni , qui démontre leur usage .

Lancement automatique de Macros et de Plugs-in Lua

En éditant dans le fichier .INI , vous pouvez vous arranger pour qu'une Macro ou plug-in Lua ou plus soient exécutées , en ordre, automatiquement selon que l'avion courant est changé (ou , en fait , le premier chargé en mémoire) ou qu'un avion nommé spécifiquement (ou un Profil) soit chargé en mémoire .

Ça permet à des switches , des offsets et à d'autres choses d'être réglés spécifiquement pour un avion (ou un type d'avions , pour des Profils) lorsqu'il est d'abord (first) chargé en mémoire .

C'est fait en ajoutant de nouvelles sections au fichier .INI avec le titre

[**Auto**] ou [**Auto.xxx**…]

Où « xxxx » est le nom de l'avion , ou une partie du nom (comme dans les sections Avions spécifiques) , ou un nom de Profil lorsque des Profils sont utilisés .

Ces sections Auto sont donc parallèles aux sections Keys et Buttons – le façon de nommer et la sélection suivent le même système .La section générique [Auto] est exécutée pour tous les changements d'avion , alors

que les sections spécifiques s'appliquent seulement à concorder avec un avion ou un profil .

Chaque section Auto contient une série de lignes numérotées (1=…,2=… etc.) dont chacune est soit une commande Lua , ou bien un appel Macro . par exemple :

[Auto.737]
1=LuaSetMyOffsets
2=737 OHD :Air Altbleeds

Lorsque Lua appelle un plug-in à se lancer , qui ne peut pas s'interrompre de lui-même , le fil (thread) du plug-in encore en train de fonctionner est tué automatiquement lors d'un changement d'avion/profil .

Assignements d'Axes

Les assignements d'Axes sont sauvegardés dans la section [Axes] , ou[Axes.<aircraft name>] pour des assignements avion-spécifique .Les assignements avion génériques peuvent être faits en utilisant les mêmes paramètre et nom abrégé que ceux des sections Buttons et Keyboard .

L'intervalle de sondage (polling) peut être changé par un paramètre :

PollInterval=10

Inséré dans la section principale [Axes] . Les unités sont les millisecondes , 10 par défaut .

Le format des paramètres axes dans ces sections est le suivant :

Pour l'entrée Axe principal (explication des valeurs ci-dessous)

n=ja,(R)delta'/delay)

où les parenthèses montrent simplement les parties optionnelles , et

j= joystick #(0 à 18 , PFC étant 16 à 18)
a = axis (XYZRUVSTPQMN)
R est présent seulement lorsque le mode « Raw » est sélectionné
Delta est la valeur delta (=512 , ou =1 pour mode Raw et POVs)

/delay est un délai optionnel * en millisecondes

Lorsque les commandes d'Axes sont assignées (la partie gauche des options) , ils sont étendus par la définition des commandes :

n=ja,(R)delta(/delay),ForD,ctl1,ctl2,ctl3,ctl4

Où :

ForD est un F pour "commande Fs" ou D pour « calibration directe vers FSUIPC »

Ctl1 à ctl4 sont les numéros de commandes , ou 0 lorsqu'elles ne sont pas assignées . Pour le Mode Direct , ce sont les indices de calibration , 1-4 à la page 1 de Calibrations , 5-8 à la page 2 , etc . Les numéros 4(-48 sont les commandes « duelles » , assimilées aux autres selon que FS est en mode flight ou en mode Slew .

ENCADRE

FSUIPC4 peut appliquer des délais à n'importe quel axe assigné à travers ses fonctionnalités d'assignements d'Axes .Le délai est limité à un minimum de 2 x l'intervalle de sondage (polling) de l'axe (10 mSecs par défaut) et un maximum de 200x cet intervalle (c'est-à-dire 2 secondes avec l'intervalle de sondage par défaut) .

Les délais pour les axes doivent être édités dans le fichier .INI . Il n'existe pas de fonctionnalité pour les changer ni même les voir dans les options écrans . Des délais de 200 mSecs ou plus devraient être raisonnablement maintenus de façon précise la plupart du temps , mais de plus courts pourraient varier un peu , et plus ils sont réglés petits plus ils sont susceptibles de varier , à cause de la granularité de l'intervalle de sondage , et le partage des tâches dans le processeur avec d'autres choses qui tournent sur FS .

Voici l'exemple d'un axe assigné au Volet de FSUIPC4 , avec un délai d'une seconde :

0=0Y,256/1000,D,22,0,0,0

Si l'axe est programmé pour envoyer des commandes basé sur un axe passant à travers des zones (le côté droit des options) il y aura aussi des entrées pour de tels assignements , ainsi :

n=ja,UDorB(R),low,high,ctl,param

où

UDorB signifie : U pour Up , D pour Down ou B pour : les deux
R veut dire , optionnellement : Repeat
low et high donnent les valeurs de l'axe pour la zone
Ctl et param sont les numéro de commandes , et de paramètre lorsqu'ils sont utilisés

Voici un exemple pour un levier de train d'atterrissage :

1=0Z,256/500
2=0Z,U,6400,16383,66079,0
3=0Z,D,-16384,-13783,66080,0

Notez que l'option délai (ici , une demi-seconde) continue encore sur l'entrée de l'axe principal , celle qui définit le délai (et le mode « raw » , s'il s'applique) .

Vous pouvez éditer le fichier .INI pendant que FS est lancé , en allant simplement à la page d'options d'Assignement des Axes et en cliquant sur le bouton reload en bas de la fenêtre .

Paramètres supplémentaires pour calibrer (scale) des valeurs d'entrées d'axes

La valeurs des axes assignées dans FSUIPC4 peuvent être ajustées arithmétiquement avant d'être passés sur la calibration de FSUIPC4 (ou vers FS via les commandes FS) . Pour ce faire , vous assignez l'axe comme normal , puis vous éditez le fichier FSUIPC4.INI . Trouvez –y l'assignement de l'axe dans la section pertinente [Axes]et ajoutez un ou deux (both) de ces paramètres à la fin :

,*<number> pour multiplier la valeur de l'axe par <number> . ça peut être une fraction , comme 0.5 (pour diviser par 2), et ça peut être négatif , pour inverser la direction de l'axe

,+<number> ou -<number>pour ajouter ou soustraire un nombre (un entier , pas de fractions) vers la valeur ou : depuis la valeur .

Si les deux paramètres sont donnés , la multiplication doit venir d'abord , et est effectuée d'abord . La valeur du résultat doit se trouver dans la gamme – 16384 à +16383 .

Par exemple si la gamme normale d'entrée d'un axe est -16384 à +16383 et que vous vouliez seulement la moitié positive , mais que vous avez besoin d'utiliser encore l'entièreté du mouvement du levier :

,*0.5,+8192

sera ajouté à l'assignement . Le µ0.5 change la gamme de -8192 à +8191 , et alors , ajouter 8192 donne 0 à 16383

Après avoir édité , dites juste à FSUIPC de recharger en mémoire les assignements d'axes (un bouton sur la page Axes) . Vous ne verrez pas les résultats là , mais dans la page Calibrations .

PROGRAMMES , FONCTIONNALITES POUR CHARGER EN MEMOIRE ET LANCER DES PROGRAMMES SUPPLEMENTAIRES

FSUIPC4 peut , à la façon d'un extra , faire en sorte que d'autres programmes puissent être lancés chaque fois que vous chargez en mémoire et que vous lancez FS . Des détails à propos de quels programmes peuvent être lancés sont fournis dans une section supplémentaire du fichier FSUIPC4.INI . Cette sections ne peut pas être éditée dans la boite de dialogue d'options en ligne de FSUIPC4 . Vous devrez éditer directement les détails dans le fichier .INI .

La section supplémentaire est :

[Programs]

Et peut contenir jusqu'à 16 requêtes pour lacer d'autres programmes – jusqu'à 8 paramètres « Run » , de Run1 à Run8, et jusqu'à 8 paramètres « RunIf », de RunIf1 à RunIf8 . Les deux réglages sont , d'autre part , identiques en format . La seule différence , c'est que les programmes RunIfne sont pas lancés s'il apparaît qu'ils sont déjà lancés. Les

programmes ordinaires « Run » seront chargés en mémoire sans de telles vérifications (checking) .

Le format est simplement :

RunN=(Options,)<full pathname of program to be run>
ou
RunIfN=(Options,)<full pathname of program to be run>

Où N va de 1 à 8 . Des détails des options sont donnés ci-dessous , mais si aucun n'est demandé , le paramètre se simplifie en juste l'intégralité du chemin de répertoire (pathname)

Par exemple : Run1=D:\RadarContact\RCV4.exe

Peut être utilisé pour lancer Radar Contact version 4 .

Si le programme nécessite des paramètres ligne de commande , ils peuvent être inclus en mettant la valeur entière entre guillemets , de telle sorte que l'espace (les espaces) dont ils ont besoin ne causent pas de problèmes . Vous pouvez également avoir besoin de mettre des guillemets si le chemin de répertoire (pathname) inclut des espaces . Par exemple /

Run2=«c:\epic\loadepic fs98jet »

Les programmes sont charges en mémoire dans l'ordre du numéro de run , de 1 à 8. Si vous avez un mélange de paramètres Run et RunIf , l'ordre est : Run1 , RunIf1 , Run2 , RunIf2 , etc .

Les options que vous pouvez utiliser sont les suivantes :

HIDE essaie d'obtenir du programme qu'il se cache lorsqu'il se lance. C'est possible seulement si le programme définit ses fenêtres de sorte qu'elles utilisent les réglages par défaut , ce n'est donc malheureusement pas très pratique pour pas mal de programmes

HIGH lance le programme avec une priorité plus haute que FS . **A utiliser avec précaution !** Se moquer des priorités ne fonctionne pas bien quelles que soient les circonstances , et FSX pourrait ne pas trop apprécier .

CLOSE ferme le programme proprement (si possible) lorsque FS est terminé

KILL termine le programme de force , si possible , lorsque FS est terminé

LOW lance le programme en priorité IDLE . Dépendant de ce que fait le programme , il peut effectivement le stopper tant que vous ne vous focaliserez pas directement sur son utilisation , puisque FS tend à absorber tout le temps Idle .

READY des délais de chargement en mémoire et de lancement du programme jusqu'à ce que FS soit ok et prêt à voler , et FSUIPC4 peut fournir des données valides à travers son interface IPC . (ce paramètre peut , bien sûr , entraîner que les programmes lancés le soient dans un ordre différent de celui spécifié par le numéro de Run) .

Seuls CLOSE , KILL et READY sont d'un usage courant . Si vous voulez appliquer plus d'une option , listez-les , séparées par des virgules , mais *pas d'espaces* . par exemple :

RunIf1=REDAY,KILL,D:\FS2002\WeatherSet.exe

Assignement de commandes d'axes supplémentaires

(inverseur , Aileron et compensateur de palonnier , capot de volets)

FS ne fournit pas de commandes d'axes pour l'inverseur de poussée des jets , pas plus que pour les ailerons ou le compensateur de palonnier ou même pour régler les capots de volets . Pour contourner ceci , et pour les assignements d'axes qui ne sont pas possibles avec les menus de FS , cochez s'il vous plaît les fonctionnalités d'assignement d'axes dans les options FSUIPC4 . vous trouverez là davantage que commandes type axes que vous n'aurez besoin qui , en dirigeant les compensateurs (trim) d'aileron , de palonnier et les capots de volets vers les propres calibrations de FSUIPC4 , peuvent être opérationnels en quelques minutes . La section Joystick de FSUIPC4 (pages 7 ou 8) traite de cela .

La commande inverseur (reverser) est spéciale , elle peut être assignée et calibrée de la même façon . De plus il y a un autre paramètre de commande :

MaxThrottgleForReverser=0

Qui commande l'enclenchement (interlock) – l'inverseur ne s'engagera pas tant que les manettes des gaz sont réduites à ce réglage (normalement 0 , ou idle) . Vous pouvez essayer une valeur non nulle si vous ne pouvez pas calibrer vos manettes des gaz pour produire un ralenti (idle) 0 stable .

Joysticks multiples pour pilotes multiples

Méthode 1

FSUIPC4 , qui utilise les sections Joystick pour calibrer les commandes principales de vol , peut accepter également jusqu'à 4 entrées commandes différentes pour chaque commande principale de vol , les traitant en parts égales . Vous pouvez avoir jusqu'à 4 commandes aileron , élévateur , palonnier , manette des gaz , freins gauche et droit . FSUIPC4 donne à la valeur d'entrée une déviation (deflection) maximale à partir de « neutre » ou de « ralenti » (idle) . Il n'y a pas de place pour des demi-mesures (averaging) ou pour d'autres types de résolution de conflit .

D'une façon ou d'une autre , vous devez connecter vos multiples axes de joystick, soit en utilisant une carte EPIC , des ports Jeux multiples ou des périphériques USB multiples . FSUIPC4 ne peut pas vous aider en la matière . Une fois cela fait , vous aurez besoin de trouver des commandes FS « disponibles » que d'autre part vous n'utiliserez pas à partir de vos entrées joystick . (voir la liste PDF des commandes FSX) . – utiliser ces commandes à partir du clavier n'a pas d'importance . FSUIPC4 ramasse (pinches) seulement les entrées joystick . Vous devez assigner les axes supplémentaires du joystick , où qu'ils puissent être , à ces commandes « disponibles » .

Maintenant , ajoutez la section JoystickCalibration du fichier FSUIPC4.INI (ajoutez la section si nécessaire) à une liste de déclarations qui définissent les commandes additionnelles que vous avez assignées . Vous les définissez *par numéro* . Les commandes de vol principales sont définies par des paramètres comme suit :

AileronB=<numéro commande>
ElévateurB=<numéro commande>

PalonnierB=<numéro commande>

D'autres paramètres peuvent définir : FreinGaucheB , FreinDroitB , ManettedesgazB , et aussi les versions C et D des 6 commandes , fournissant ainsi jusqu'à 4 copies de chacune d'entre elles .

Notez que vous aurez besoin de calibrer toutes les commandes de telle sorte que celles qui commandent les mêmes valeurs soient aussi proche de possible en gamme (range) et en réponse . faites cela d'abord dans le Panneau de Commande Windows , puis , après faites les ajustements ci-dessus et les assignement , dans FSUIPC4 . Calibrez les zones mortes (dead zones) aux extrémités (et au centre pour l'aileron , l'élévateur et le palonnier) pour dissimuler tout « décalage »- en d'autres mots calibrez en tenant compte du pire pour chacune .

Méthode 2

Une méthode plus facile est maintenant disponible , vous permettant d'utiliser la fonctionnalité d'Assignement d'Axes de FSUIPC pour assigner vos commandes , après avoir détruit (deleting) les assignements FS .

Les assignements d'axes de FSUIPC permettent à n'importe quel axe de vos joysticks d'être assigné à n'importe quelle commande d'axe de FS ou de FSUIPC, et il n'y a aucune restriction quant à combien vous pouvez en assigner à combien d'entre eux . Donc le premier problème étant résolu , vous pouvez assigner 2 yokes , palonniers , quoi qu'il en soit , aux mêmes commandes .

FSUIPC et FS tiennent compte du dernier mouvement dans un axe . Ils ne le « sondent » pas (poll) pour obtenir des entrées régulières , mais ils voient seulement les changements qui en arrivent . Tous deux verront le dernier changement depuis des axes multiples . Cependant ,n ce peut être un léger mouvement accidentel ou une vibration (jitter) non désirée. De la sorte , en vous permettant d'assigner directement vos axes avec la calibration FSUIPC (en tant qu'opposée à une commande FS) , c'est FSUIPC qui maintenant arbitre , sélectionnant les axes avec la déviations la plus grande (définie ici comme une différence à partir de 0)

Notez , cependant , qu'il voit encore les axes lorsqu'ils changent , ainsi même lorsqu'un axe est maintenu dans une grande déviation , une fois qu'un autre axe , pour la même commande ,se déplace à une position

similaire ou plus haute , ce dernier prend alors la commande même s'il se déplace plus lentement que le premier- le dernier est effectivement « out » jusqu'à ce qu'il soit déplacé .

Les conseils au sujet de la calibration de la Méthode 1 s'appliquent encore .

FONCTIONNALITES COMPENSATEUR DE TANGAGE ET ROULIS - HELICOPTERES

Une fonctionnalité pour faire fonctionner le compensateur de tangage et roulis des hélicoptères est fournie . Elle utilise les commandes normales FS des compensateur d'élévateur et d'aileron pour modifier la valeur terminale des axes Y (élévateur) et X (aileron) du cyclique (cyclic) . Pour l'utiliser , vous aurez besoin de vous assurer que les axes sont calibrés en utilisant FSUIPC4 (respectivement comme axes de l'élévateur et de l'aileron) . ; et ajoutez

ApplyHeloTrim=Both

à la (aux) section(s) pertinente(s) [JoystickCalibration …] dans le fichier FSUIPC4.INI . Notez que , par précaution , la valeur du compensateur (trim) ne sera jamais ajoutée à l'axe dont il relève si la valeur normale du compensateur n'est pas égale à zéro .

Ces nouvelles valeurs « helo trim » sont maintenues dans les offsets IPC comme suit :

0BBE 2 bytes 16-bit Helo Pitch Trim value, range-16383 to +16383

0C06 2 bytes 16-bit Helo Bank Trim value, range-16383 to +16383

Les deux peuvent être écrites pour une commande de programme externe .

Notez que si vous demandez seulement un compensateur de tangage (pitch trim) vous pouvez écrire :

ApplyHeloTrim=Yes

A la place de « both » Les valeurs d'axes et de compensateur (aileron/tangage resteront seules .

FILTRES MESSAGES

L'affichage des messages envoyés à FSUIPC sur l'écran FS peuvent être filtrés et acheminés de force en fonction de leurs premiers caractères . Pour ce faire on ajoute comme suit une nouvelle section au fichier FSUIPC.INI :

[Message Filters]
Suppres=…
SingleLine=…
MultiLine=…

La partie « … » est remplacée par une liste qui va jusqu'à 8 chaînes (strings) (entre guillemets) , chacune de moins de 16 caractères . Les messages envoyés à FSUIPC sont comparés avec elles . S'ils démarrent avec les mêmes caractères (cas ignoré) alors l'action prise est comme suit :

Suppress : le message est abandonné
SingleLine : le message est traité comme un message simple-ligne , même s'il ne l'est pas
MultiLine : le message est traité comme un message multi-ligne , même s'il ne l'est pas .

Par exemple :

SingleLine= »FDC », »PM MCP »

Acheminera les messages commençant par « FDC » ou « PM MCP » à une fenêtre simple-ligne , à moins que de tels messages soient supprimés par l'option FSUIPC .

FICHIERS INI MULTIPLES

En général , à cause des fonctionnalités d'assignement et de calibration des appuis touches clavier , axes et boutons d'avions-spécifiques divers , construits dans FSUIPC4 , il n'est pas réellement souvent nécessaire de penser qu'il faille différents fichiers INI pour différents besoins . Mais c'est possible . Voici comment :

Pour chaque set supplémentaire de paramètres INI , vous devez faire une copie du fichier FSX.EXE dans le répertoire principal FSX , avec un autre nom . Il existe deux variations sur la façon dont on utilise le nom changé .

Si vous donnez un nom qui commence par « FSX » , comme **FSX_for_Choppers.EXE** , alors FSUIPC4 utilisera la partie jointe (appended) pour ses fichiers , ainsi :

FSUIPC4_for_Choppers.INI

Et ainsi pour **.LOG** et même **.KEY** – donc souvenez-vous de dupliquer votre fichier KEY et renommez-le en conséquence ou vous vous trouverez vous-même non enregistré) .

Si vous ne gardez pas « FSX » comme premiers caractères , alors l'intégralité du nouveau nom est joint après un « . »- par exemple un **PetesFSX.EXE** donnerait :

FSUIPC4.PetesFSX.INI (et , de même , la même chose pour **.LOG** et **.KEY**) .

IMPORTANT : Chaque FSX renommé a également son fichier .CFG renommé – si vous ne voulez pas copier votre fichier FSX.CFG et le renommer pour qu'il s'assortisse à votre fichier FSX.EXE , lorsque FSX se rechargera en mémoire il en génèrera un nouveau , par défaut , avec le nouveau nom .

ANNEXE 1 : FAITES PLUS AVEC VOTRE JOYSTICK !

Cette section vient d'une contribution faite gracieusement par un utilisateur intrépide de FSUIPC . J'espère que vous la trouverez utile . Si ce n'est formater pour adapter ce document , je l'ai laissé exactement dans l'état où il m'a été soumis .

Durant les dernières années de simulation de vol , les simulateurs de vol pour PC sont devenus de plus en plus professionnels et de plus en plus complexes . Des avions très sophistiqués peuvent être téléchargés gratuitement ou pour une somme modique . Beaucoup d'entre eux incluent tous les accessoires en ce sens que la simulation de vol n'est plus du tout un « jeu » et , pour beaucoup d'entre nous , il devient un « simulateur de vol réel » utilisé comme tel dans le monde réel de l'aviation . Quelques uns ont construits des cockpits très proches de la réalité avec des tableaux de bord où les interrupteurs et commandes d'instruments sont à leurs véritables emplacement ; d'autres , comme moi-même , utilisent toujours leur joystick et leur clavier .

Comme je possède un petit ordinateur de bureau , il n'est pas si pratique d'utiliser ensemble mon clavier et mon joystick , spécialement pour la simulation de vol . Un cockpit est surchargé de périphériques à installer , de nombreux interrupteurs doivent être utilisés , beaucoup de réglages doivent être effectués à partir d'entrées clavier et avec un malheureux (scrubby) joystick à 8 boutons pour toutes les commandes restantes , il semble impossible de le faire d'une façon qui soit sympa pour l'utilisateur .

Dans FSX et les simulateurs précédents , Microsoft a assigné quelques commandes par défaut (mais pas toutes) aux boutons de joystick . Celles-ci peuvent être modifiées par le menu3 « Options-Controls-Assignments » . En sélectionnant une commande à partir d'une liste et en définissant un bouton de votre joystick , l'activation de ce bouton activera la commande sélectionnée . Il existe aussi une option qui répète la commande aussi longtemps que le bouton est pressé .

Je possède un joystick 8 boutons , mais la simple programmation par défaut des 8 boutons n'était pas suffisante . J'avais besoin de plus de commandes , de davantage de sophistication sur mon joystick . J'ai donc

cherché une solution : parce que mon joystick était absolument nécessaire la solution fut alors d'éliminer autant que possible les entrées clavier .

Par chance il existe encore de par ce monde des gars sympas , des types comme Pete Dowson . Pete Dowson est bien connu pour son excellent module add-on FSUIPC.DLL pour MS FS . Ce module permet de corriger quelques failles dans FS et de l'améliorer , il doit être considéré comme un « must » pour la communauté FS toute entière . Mais FSUIPC inclut également beaucoup de fonctionnalités que l'utilisateur lambda (modal) peut utiliser à son avantage. L'une d'entre elles , qui s'adresse aux utilisateurs qui possèdent la licence FSUIPC , s'intitule « programmation bouton joystick et clavier » .

A l'origine , Pete a fourni cette fonctionnalité aux possesseurs de périphériques Goflight et Epic , mais elle peut également être utilisée pour votre joystick ! J'ai écrit ce Guide à la demande de Pete lorsque nous avons pris conscience que seuls quelques rares utilisateurs FS utilisaient ce puissant outil à bon escient .J'essaierai de vous expliquer les choses merveilleuses que vous pouvez faire avec ce superbe outils de programmation . Il y a quelques semaines je ne le réalisais pas encore moi-même , mais depuis , oh boy ! …

J'expliquerai quelques trucs de programmation que j'utilise dans la programmation de mon propre Microsoft Joystick2 Sidewinder Force Feedback .

La documentation qui suit est nécessaire avant de démarrer la programmation :

1°) une licence utilisateur complète de FSUIPC4 , version 4 , installée dans votre copie de FSX .

2°) le manuel « FSUIPC4 pour Utilisateurs Avancés » [*celui qui est maintenant devant vous*] Lisez s'il vous plaît très soigneusement les chapitres concernant la programmation des boutons joystick et clavier , particulièrement la section sur les boutons-composés .

3°) le document « Liste des commandes FSX » qui aura aussi été installé pour vous dans votre répertoire Modules de FSX .

Pour être sûr que les commandes s'exécuteront bien de la façon dont vous les avez programmées , presque toutes les programmations de

bouton par défaut dans FS doivent être enlevées. Je les ai toutes enlevées , excepté la programmation du bouton Hat .

J'utilise les 2 boutons sur la gauche du pied du joystick pour régler les conditions de sélection des commandes assignées aux 6 autres boutons . J'inclus également les trucs que vous pouvez utiliser au cas où vous voudriez utiliser 3 boutons pour régler les conditions .

Commençons avec les premier cas . Les 2 boutons pour définir une condition sont étiquetées « 7 » et « 8 » sur le joystick . L'étiquette la plus basse sur le joystick est « 1 » . Cependant , pour programmer , la numérotation de bouton commence avec le bouton « 0 » pour le bouton qui a l'étiquette 1 , « 1 » pour le bouton 2 et ainsi de suite . Et donc , dans notre cas , les conditions sont programmées par les boutons « 6 » et « 7 » . Le statut des boutons permet 4 possibilités , up ou down , une combinaison de deux boutons :

1. boutons 6 et 7 sont tous deux up
2. bouton 6 down bouton7 up
3. bouton7 down bouton6 up
4. bouton6 et bouton7 sont tous deux down

Le statut de ces deux boutons : l'action de l'un deux sur les autres 6 boutons peut être utilisé pour programmer une commande de simulateur de vol . En fait nous pouvons désormais assigner jusqu'à 4 commandes par bouton ou 24 commandes aux 6 boutons restants (même 48 , parce que nous pouvons programmer une fonction si l'un des boutons actions est down et une autre fonction quand le même bouton est de nouveau up) .

Ce qui suit peut être fait avec une combinaison de 3 boutons :

1. boutons 5,6 et 7 tous les 3 up
2. bouton5 down , 6 et 7 up
3. bouton6 down , 5 et 7 up
4. bouton7 down , 5 et 6 up
5. boutons 5 et 6 down , bouton 7 up
6. boutons 5 et 7 down , bouton 6 up
7. boutons 6 et 7 down , bouton 5 up
8. boutons 5 , 6 et 7 down

Il y a maintenant dans la combinaison 8 possibilités et on pourra assigner de 40 à 80 commandes aux 5 boutons restants .

Comme spécifié dans la section sur la Programmation de Bouton , ci-avant , 2 sortes de commandes peuvent être générées : utiliser la combinaison de bouton pour simuler l'appui d'une combinaison de touches sur le clavier , ou bien utiliser la combinaison de boutons joystick pour générer une commande « interne » FS . Une liste de toutes les possibilités peut être trouvée dans la « Liste des commandes FSX » .
Prenons comme exemples quelques règles de programmation de bouton :

3=CP(-0,6)(-0,7)0,3,C65615,0
…
…
9=CP(+0,6)(-0,7)0,3,C65769,0

Dans ces deux cas , le bouton actif (celui qui génère la commande) est le bouton 3 (avec l'étiquette 4 sur le joystick) . dans le premier cas , la commande « 65615 » est générée lorsque les boutons 6 et 7 sont up et que le bouton 3 est en train d'être down . C65615 génèrera un « Elevator trim up » (compensateur d'élévateur activé) ,la même commande que la programmation par défaut du bouton de joystick . La syntaxe « CP » définit que la commande sera exécutée une fois seulement , même si le bouton 3 est maintenu down .

Cependant , en appuyant sur le bouton « 6 » (down) (« 7 » sur le joystick) et en activant le bouton 3 , FSUIPC génèrera une commande « 65769 , Propeller Pitch increment » (augmentation pas d'hélice). Cette commande n'est pas une commande de bouton de joystick par défaut, mais une commande qui , si elle n'était pas programmée de cette façon , devrait être entrée par une combinaison de boutons sur le clavier .

En définissant la combinaison de bouton avec un « CR » , la commande sera répétée jusqu'à ce que le bouton soit relâché à nouveau , ce qui est , dans notre application , plus avantageux . Et , en fait , la fonction repeat est utilisée sur les deux commandes :

2=CR(-0,6)(-0,7)0,2,C65607,0
3=CR(-0,6)(-0,7)0,3,C65615,0
…
…
8=CR(+0,6)(-0,7)0,2,C65771,0
9=CR(+0,6)(-0,7)0,3,C65769,0

Les boutons « 2 » et »3 » sont utilisés ici pour le compensateur (trim) up/down (règles 2 et 3) avec les boutons 6 et 7 up. Les mêmes boutons , mais maintenant avec le bouton 6 activé pendant que le bouton 7 est up , commandent le pas d'hélice .

J'ai assigné deux autres commandes aux boutons « 2 » et »3 », j'ai programmé la combinaison avec le bouton 7pour « Mixture Incr » et « Mixture Decr » dans les règles 19 et 20 :

19=CR(-0,6)(+0,7)0,2,C65775,0
20=CR(-0,6)(+0,7)0,3,C65777,0

Je dois insister ici sur le fait que FSUIPC utilise le *statut* (up ou down) des boutons dans les combinaisons composées (+/-j,b)(+/-j,b) en tant que condition , mais utilise les *changements* de statuts du bouton (en fait le « pousser » ou le « relâcher » du bouton de joystick) en tant qu'activation de la commande , qui est valide pour un balayage (scan) entier , ce qui signifie la vérification de toutes les règles de programmation de bouton . Il est important de se le rappeler .

Sur les joysticks MS , les boutons 2 et3 sont très bien placés pour qu'on les utilise comme fonction d'augmenter et de diminuer et beaucoup de commandes pourraient leur être attachées . Cependant , nous avons déjà utilisé 3 des 4 statuts de condition . Donc si nous utilisons seulement la combinaison de deux boutons et que nous aimerions attacher davantage de commandes à ces boutons , nous devons trouver une autre manière de procéder .

En tout premier lieu , comme toutes les versions précédentes , les mises à jour incluant la version 3.14 , FSUIPC permet des combinaisons composées du statut jusqu'à deux boutons , pour créer une condition . Dans les versions ultérieures , le statut de 16 boutons peut être utilisé pour créer une condition , mais les astuces expliquées seront encore valides .

En fait , la programmation de bouton ne travaille pas directement avec les boutons parce que FSUIPC entrepose le statut d'un bouton dans un « flag » (drapeau), un espace de stockage interne , durant un cycle de processeur ou bien un scan de toutes les règles de bouton programmées .

Désormais , FSUIPC sauvegarde le statut de jusqu'à 32 boutons de jusqu'à 16 joysticks , ce qui signifie 512 flags pour 512 boutons !

De ces 512 flags , seulement 8 sont utilisés pour les 8 boutons de mon joystick et le reste de ces flags semble être de l'espace gaspillé . Pas tout à fait ! Parce que Peter a fourni quelques commandes pour installer (set) switcher (toggle) ou réinstaller (reset) les flags , même s'ils ne sont pas « connectés » à un bouton . Donc une instruction peut être utilisée pour installer (set) ou réinstaller (reset) un flag et pour utiliser le flag par la suite dans une condition . Et parce qu'il n'y a pas de connexion à un bouton existant , le statut du flag est entièrement dépendant des instructions programmées qui lui sont données .

Maintenant nous allons créer UN flag reflétant les conditions de DEUX boutons, de telle sorte que ce flag puisse être utilisé avec le statut d'un autre bouton , pour créer une autre condition . Pour ce faire j'utilise les astuces suivantes :

```
;Flag 10 follows keys (+6 AND -7)
;
0=CU(-0,7)0,6,C1003,10
1=CU(-0,6)0,7,C1003,10
```

Lorsque FS est démarré et que le module FSUIPC.DLL est activé , tous les drapeaux sont réinstallés (reset) . Pour être sûrs de l'installation du flag 10 , nous devons « jouer un bit » avec les boutons 6 et 7 . Jouer un bit avec ces boutons au début de notre vol ne génère pas de commandes , parce que les deux boutons sont des boutons « morts » qui n'enverront pas de commandes à FS (c'est la même chose que les touches shift qui ne font rien par elles mêmes , mais qui fonctionnent toujours avec d'autres touches). Les règles ci-dessus assurent que le flag 10 lorsque les deux boutons seront up ou seront en train d'être up :

Règle 0 : quand le bouton 7 est up et que le bouton 6 va sur up , alors installer Flag 10

Règle 1 : quand le bouton 6 est up et que le bouton 7 va sur up , alors installer Flag 10

Les règles suivantes installent des flags lorsque l'un des deux boutons est en train d'aller sur down. Cependant dans ces cas , nous devons réinstaller flag 10 :

```
;Flag 11 follows keys (+6 AND -7)
;
2=CP(-0,7)0,6,C1004,10
```

```
3=CP(-0,7)0,6,C1003,11
4=CU(F+0,11)0,6,C1004,11
;
;Flag 12 follows key (-6 AND +7)
;
5=CP(-0,6)0,7,C1004,10
6=CP(-0,6)0,7,C1003,12
7=CU(F+0,12)0,7,C1004,12
```

L'explication de ces règles de programmation est :

Règle 2 : si bouton 6 va sur down et que bouton 7 est encore up , réinstaller flag10

Règle 3 : si bouton 6 va sur down et que bouton 7 est encore up , installer flag 11 (se rappeler que l'action du bouton actif peut être vue par toutes les règles qui suivent dans le même scan) .

Règle 4 :quand flag 11 est installé et que bouton 6 va sur up , réinstaller flag 11

Flag 11 suit maintenant le statut du bouton 6 (up ou down) pendant que bouton 7 est up .

Règle 5 : si bouton 7 va sur down et que bouton 6 est toujours sur up , réinstaller flag 10

Règle 6 : si bouton 7 va sur down et que bouton 6 est toujours sur up , installer flag 12

Règle 7 : si flag 12 est installé et que bouton 7 va sur up , réinstaller flag 12
Dans ce cas , flag 12 suit le statut de bouton 7 (up ou down) pendant que bouton 6 est up .

Voici maintenant une partie plus astucieuse parce que nous voulons créer un « suiveur » (follower) pour les boutons 6 et 7 down(si nous ne voulions pas utiliser une combinaison avec les deux boutons down , alors nous devrions inclure dans tous les cas les règles 8 et 11 pour nous assurer de la réinstallation des flags 11 et 12 lorsque les conditions des règles ci-dessus ne sont plus valides) :

```
;flag 13 follows key ( +6 et +7 )
```

```
;
8=CP(+0,6)(F+0,11)0,7,C1004,11
9=CP(+0,6)0,7,C1003,13
10=CU(F+0,13)0,7,C1004,13
;
11=CP(+0,7)(F+0,12)0,6,C1004,12
12=CP(+0,7)0,6,C1003,13
13=CU(F+0,13)0,6,C1004,13
```

Même si nous faisons de notre mieux , il est presque impossible de pousser deux boutons en même temps , nous devons donc infirmer (disable) le flag résultant de l'installation des règles 8 et 11parce que la bouclé programmée détectera que les conditions spécifiées dans les règles 3 ou 6 seront satisfaites avant que le second bouton soit activé :

Règle 8 : si bouton 6 est down et que flag 11 est installé (parce que nous sommes allés plus vite avec le bouton 6 qu'avec le bouton 7) et que le bouton 7 va sur down , réinstaller flag 11

Règle 9 : si bouton 6 est down et que bouton 7 est down , installer flag 13

Règle 10 : si flag 13 est installé et que bouton 7 est relâché , réinstaller flag 13 . Cette règle de programmation agit si bouton 7 est relâché avant bouton 6 . Dans ce cas , vous pourriez penser que la règle 3 est de retour dans le jeu , mais ce n'est pas vrai : FSUIPC ne réagit pas au statut de la touche « active » mais au changement de son statut : « bouton down » signifie actuellement « bouton va sur down », bouton up signifie actuellement : « bouton va sur up » . Et , parce qu'il n'y a pas de changement dans le statut de bouton 6 , la règle 3 n'est pas activée .

Règle11 : si bouton 7 est down et que flag 12 est installé (parce que nous sommes allés plus vite avec bouton 7 qu'avec bouton 6) et que bouton 6 va sur down , réinstaller flag 12

Règle 12 : si bouton 7 est down et que bouton 6 va sur down , installer flag 13

Règle 13 : si flag 13 est installé et que bouton 6 est relâché , réinstaller flag 13 . Cette règle de programmation agit si bouton 6 est relâché avant bouton 7 . Même remarque ici que pour la règle 10 , mais qui concerne la règle 6 .

Désormais , ces flags peuvent être utilisés pour assigner de vraies fonctions FS aux boutons restants :

```
;IF -6 AND -7 (Flag10 )
;
14=CR(F+0,10)0,0,C65588,0              ;répète freins
15=CP(F+0,10)0,1,C65570,0              ;toggle gear
16=CR(F+0,10)0,2,C65607,0              ;repeat trim pitch up
17=CR(F+0,10)0,3,C65615,0              ;repeat trim pitch down
18=CP(F+0,10)0,4,C65758,0              ;increment flaps
19=CR(F+0,10)0,5,C65759,0              ;decrement flaps
;
;IF +6 AND -7 (Flag11)
;
20=CP(F+0,11)0,0,K192,1                ;voice key for CS727
21=CP(F+0,11)0,1,C65751,0              ;toggle landing lights
22=CR(F+0,11)0,2,C65771,0              ;repeat decr. mixture
23=CR(F+0,11)0,3,C65769,0              ;repeat incr. mixture
24=CP(F+0,11)(F+0,20)0,4,C66390,0      ;AND F+20 toggle wing fold
25=CP(F+0,11)(F-0,20)0,4,C66391,0      ;toggle tail hook
26=CP(F+0,11)0,5,C65589,0              ;toggle air break
```

Commentaire supplémentaire ligne 24 : (Drag chute on CSF104)
Commentaire supplémentaire ligne 25 (Drag chute on CSMig21)

Un petit peu plus au sujet des règles 24 et 25 : Je suis fan des avions Captain Sim mais l'équipe CS utilise une commande pour le largage de parachute sur le Mig 21différente de celle utilisée pour le Starfighter . J'ai décidé d'utiliser un flag (que je programmerai plus tard) pour générer une commande différente pour le même bouton, en fonction du statut de ce flag . Voici une autre astuce que j'utilise pour le Yak 3 de Captain Sim : l'animation par défaut du rétroviseur (rear-view mirror) utilise deux commandes différentes pour l'activation et la désactivation du miroir . C'est assez bizarre parce que c'est une commande interrupteur à bascule (toggle) . L'astuce suivante me permet d'assumer la fonction toggle avec un seul bouton :

```
36=CP(F+0,13)(F-0,30)0,0,C66294,0
37= CP(F+0,13)(F+0,30)0,0,C66295,0
38=CU(F+0,13)0,0,C1005,30
```

La combinaison de ces 3 lignes (rows) est utiliser pour switcher le bouton de commande de incr vers decr et réciproquement , à chaque

fois le bouton 0 est active pendant que les boutons 6 et 7 sont tous deux down .

Les règles suivantes de programmation de bouton dans le fichier INI sont :

```
;IF (-6 AND +7) = F12
;
27=CP(F+0,12)0,1,C65858,0              ;toggle pilot heat
28=CR(F+0,12)0,2,C65777,0              ;repeat mixture decr
29=CR(F+0,12)0,3,C65775,0              ;repeat mixture incr
30=CP(F+0,12)0,4,K83,8                 ;keyboard “S” ( next view)
31=CP(F+0,12)0,5,K83,1           ;keyboard “SHIFT-S” ( previous view )
;
;IF (+7 AND +8) = F13
;
32=CP(F+0,13)0,0,C66224,0              ;autostart engines
33=CP(F+0,13)0,1,C66293,0              ;toggle avionics on/off
34=CR(F+0,13)0,2,C65880,0              ;increment heading bug
35=CR(F+0,13)0,3,C65879,0              ;decrement heading bug
```

En utilisant la combinaison de 3 boutons , nous pouvons créer ce qui suit :

```
;Flag 10 follows keys (65 AND -6 AND -7)
;
0=CU(-0,6)(-0,7)0,5,C1003,10
1=CU(-0,5)(-0,7)0,6,C1003,10
2=CU(-0,5)(-0,6)0,7,C1003,10
;
;Flag11 follows keys (+5 AND -6 AND -7 )
;
3=CP(-0,6)(-0,7)0,5,C1004,10
4=CP(-0,6)(-0,7)0,5,C1003,11
5=CU(F+0,11)0,5,C1004,11
;
;Flag12 follows key (-5 AND +6 AND -7)
;
6=CP(-0,5)(-0,7)0,6,C1004,10
7=CP(-0,5)(-0,7)0,6,C1003,12
8=CU(F+0,12)0,6,C1004,12
;
;Flag14 follows key (-5 AND -6 AND +7)
```

```
;
9=CP(-0,5)(-0,6)0,7,C1004,10
10=CP(-0,5)(-0,6)0,7,C1003,14
11=CU(F+0,14)0,7,C1004,14
;
;Flag14 follows key (+5 AND +6 AND -7 )
;
12=CP(+0,6)(F+0,12)0,5,C1004,12
13=CP(+0,6)(-0,7)0,5,C1003,13
14=CU(F+0,13)0,5,C1004,13
;
15=CP(+0,5)(F+0,11)0,6,C1004,11
16=CP(+0,5)(-0,7)0,6,C1003,13
17=CU(F+0,13)0,6,C1004,14
;
;Flag 15 follows key ( +5 AND -6 AND +7 )
;
18=CP(+0,6)(F+0,13)0,5,C1004,13
19=CP(+0,7)(-0,6)0,5,C1003,15
20=CU(F+0,15)0,5,C1004,15
;
21=CP(+0,5)(F+0,13)0,7,C1004,11
22=CP(+0,5)(-0,6)0,7,C1003,15
23=CU(F+0,16)0,7,C1004,15
;
;Flag 16 follows key (-5 AND +6 AND+7 )
;
18=CP(+0,6)(F+0,12)0,7,C1004,12
19=CP(+0,7)(-0,5)0,7,C1003,16
20=CU(F+0,16)0,7,C1004,16
;
21=CP(+0,7)(F+0,13)0,6,C1004,13
22=CP(+0,7)(-0,5)0,6,C1003,16
23=CU(F+0,16)0,6,C1004,16
```

ANNEXE 2 à propos de l'option *Aircraft Specific* et de "*ShortAircraftNameOK*"

Note : il s'agit de la contribution d'un Utilisateur , qu'il en soit remercié .

Il y a ces 3 choix dans les réglages FSUIPC :

ShortAircraftNameOK=No

ShortAircraftNameOK=Yes

ShortAircraftNameOK=Substring

Conséquence : obtenir exactement les mêmes réglages pour AXES , BUTTONS , KEYS et CALIBRATION pour chaque avion repeint ou différent (variant) .

Dans FSUIPC , le Short Aircraft Name réfère au nom dans le fichier Aircraft.cfg sous la rubrique « title »

Par exemple : Aerosoft DHC Beaver . Il peut y avoir 7 avions différents ou repeints

aircraft.cfg \(flightsim.X)\title= Aerosoft Beaver DHC-2A 55-0682
aircraft.cfg \(flightsim.X)\title= DHC-2A C-GSKY Beaver
aircraft.cfg \(flightsim.X)\title= Aerosoft DHC-2A C-GSKY modern
aircraft.cfg \(flightsim.X)\title= Beaver DHC-2A DQ-GEE
aircraft.cfg \(flightsim.X)\title= DHC-2A DQ-GEE modern
aircraft.cfg \(flightsim.X)\title= Aerosoft DHC-2A N299EE
aircraft.cfg \(flightsim.X)\title= Aerosoft DHC-2A N299EE modern

Editer le fichier FSUIPC.ini :

Scenario 1 : if « shortnameaircraftOK=No »

En supposant que vous avez déjà assigné les axes , keys et buttons et calibré le joystick pour un des avions différents ou repeints ci-dessus : de sorte à obtenir les mêmes réglages pour le reste des Aerosoft Beaver différents ou repeints vous aurez besoin d'éditer le fichier FSUIPC.Ini et d'ajouter 4 entrées séparées pour chacun des 4 noms titre (exactement comme ci-dessus) pour [Axes] , [Buttons] , [Keys],[Joystick Calibration]

pour vous assurer que tous les réglages sont exactement les mêmes , c’est-à-dire 28 entrées en tout . . Assez fastidieux en fait – je possède plus de 40 variantes et avions repeints de cet avion , donc je pourrais avoir besoin de 160 entrées dans le fichier FSUIPC.ini .

[Axes. Aerosoft Beaver DHC-2A 55-068] [Buttons. Aerosoft Beaver DHC-2A 55-068] [Keys. Aerosoft Beaver DHC-2A 55-068] [JoystickCalibration. Aerosoft Beaver DHC-2A 55-068]
[Axes. DHC-2A C-GSKY Beaver] [Buttons. DHC-2A C-GSKY Beaver] [Keys. DHC-2A C-GSKY Beaver] [JoystickCalibration. DHC-2A C-GSKY Beaver]
[Axes. Aerosoft DHC-2A C-GSKY modern] [Buttons. Aerosoft DHC-2A C-GSKY modern] [Keys. Aerosoft DHC-2A C-GSKY modern] [JoystickCalibration. Aerosoft DHC-2A C-GSKY modern]
[Axes. Beaver DHC-2A DQ-GEE] [Buttons. Beaver DHC-2A DQ-GEE] [Keys. Beaver DHC-2A DQ-GEE] [JoystickCalibration. Beaver DHC-2A DQ-GEE]
[Axes. DHC-2A DQ-GEE modern] [Buttons. DHC-2A DQ-GEE modern] [Keys. DHC-2A DQ-GEE modern] [JoystickCalibration. DHC-2A DQ-GEE modern]
[Axes. Aerosoft DHC-2A N299EE] [Buttons. Aerosoft DHC-2A N299EE] [Keys. Aerosoft DHC-2A N299EE] [JoystickCalibration. Aerosoft DHC-2A N299EE]
[Axes. Beaver Aerosoft DHC-2A N299EE modern] [Buttons. Beaver Aerosoft DHC-2A N299EE modern] [Keys. Beaver Aerosoft DHC-2A N299EE modern] [JoystickCalibration. Beaver Aerosoft DHC-2A N299EE modern]

Scénario 2 : si “ShortAircraftNameOK=Yes”

12 entrées seront requises pour s’assurer que tous les réglages sont les mêmes .

[Axes.Aerosoft] [Buttons.Aerosoft] [Keys.Aerosoft] [JoystickCalibration.Aerosoft]	[Axes.DHC] [Buttons.DHC] [Keys.DHC] [JoystickCalibration.DHC	[Axes.Beaver] [Buttons.Beaver] [Keys.Beaver] [Joystickcalibration.Beaver]

Explication :

1. Aérosoft choisirait toutes les entrées dans le titre STARTING avec “AEROSOFT” , mais PAS Aérosoft dans toute autre partie du titre
2. « DHC » choisirait toutes les entrées dans le titre STARTING avec « DHC » mais pas DHC dans toute autre partie du titre
3. « Beaver » choisirait toutes les entrées dans le titre STARTING , mais pas Beaver dans toute autre partie du titre .

Scénario 3 : si « ShortAircraftNameOK »=Substring »

4 entrées seulement . C’est-à-dire que « DHC » dans le fichier FSUIPC.ini consisterait en tous les avions différents qui ont exactement les mêmes réglages – « DHC » est commun à tous les titres .

[Axes.DHC] [Buttons.DHC] [Keys.DHC] [JoystickCalibration.DHC]

Pour résumer :

ShortAircraftNameOK=No	**1entrée pour chaque titre différent dans le fichier aircraft.cfg**
ShortAircraftNameOK=Yes	**Choisit la partie starting du titre fans le fichier aircraft.cfg**
ShortAircraftNameOK=Substring	**Choisit n'importe quelle partie du titre dans le fichier aircraft.cfg**

Titre dans le fichier aircraft.cfg	**ShortAircraftNameOK=**		
	No	**Yes**	**Substring**
Title=Airbus A321 Title=Airbus A321 Paint2 Title=Airbus A321 Paint4 Title=Airbus A321 Paint5 Title=Boeing 737-400 Title=Boeing 737-400 Paint1 Title=Boeing 737-400 Paint2 Title=Boeing 737-400 Paint3 Title=Boeing 737-400 Paint4 Title=Boeing 747-400 Title=Boeing 747-400 Paint1 Title=Boeing 747-400 Paint2 Title=Boeing 747-400 Paint3	**Entrée séparée pour chaque titre**	**« Airbus » : s'appliquerait à toutes les entrées démarrant avec Airbus.** **« Boeing » s'appliquerait à toutes les entrées démarrant avec Boeing**	**« A321 » : tout avion différent avec A321 dans le titre** **« Paint » tout avion différent avec PAINT dans le titre** **« 737 » : tout avion différent avec 737 dans le titre**

Title=Boeing 777-300 Title=Boeing 777-300 Paint1 Title=Boeing 777-300 Paint2 Title=Boeing 777-300 Paint3			

Explication :**ShortAircraftnameOK=Substring**

Tout texte , dans n'importe quelle position dans le « titre »localisé dans le fichier aircraft.cfg , inséré dans le fichier .INI comme ci-dessus , entraînerait les mêmes réglages pour les avions . Par exemple , choisir « 737 » , c'est-à-dire [Axes.737] etc. résulterait en : tous les avions qui ont 737 dans le titre auront les mêmes réglages .De la même manière , choisir « Boeing » couvrirait tous les avions différents/repeints avec Boeing dans le titre .

Pour résumer , si vous avez 20 appareils différents/modèles/repeints avec des titres différents , vous aurez besoin de 20 entrées par section (80 en tout) dans le fichier .INI.
Utiliser **ShortAircraftNameOK=Substring** permet de les ramener à juste une entrée par section (4 en tout) .

ANNEXE 3 : prise en main des périphériques VRInsight avec FSUIPC4 .

Introduction

Les périphériques VRInsight sont devenus populaires , d'un bon rapport qualité/prix . Vous pouvez obtenir beaucoup de fonctionnalités dans un package compact . Toutefois ils ne sont pas reconnus par Windows comme « Human Interface Devices » (HIDs) et certainement pas en tant que « joysticks », et normalement , ils ne sont donc pas vus par FSUIPC en tant que boutons programmés ou switchs programmés .

En fait il s'agit de périphériques PORT COM série, qui utilisent des connexions USB avec une interface à base de chip FTDI avec un pilote PORT SERIE/USB . Leur interface à FS est managée par le propre pilote de VRInsight : « **SerialFP2** »

Pour la plupart des utilisations simples , SerialFP2 fait du bon boulot. Cependant , il ne fournit pas la visibilité pour chaque objectif et , avec de plus en plus d'avions spécialisés et autres add-ons pour FS faisant leur propre cuisine , le besoin d'une méthode qui augmente la fonctionnalité des périphériques VRI s'est fait sentir . C'est particulièrement le cas lorsque les périphériques doivent recourir à l'envoi de plusieurs combinaisons d'appuis touches clavier , ce qui peut créer une situation tendue alors même que d'autres programmes font la même chose en même temps .

L'opportunité de prévoir dans FSUIPC pour la gamme VRInsight s'est fait jour après l'implémentation du port série de maintenance (handling) de la Librairie Lua , « **com** » , parce que FSUIPC contenait déjà un mécanisme multi-périphériques multi-threads pour lire facilement à partir de , et écrire à des périphériques PORT SERIE COM .

Problèmes et Solutions

Si on la compare à l'implémentation de GoFlight dans FSUIPC, qui utilise un module librairie (GFDEv.dll) fourni par GoFlight dans ce but , il y a quelques complications . Avec GoFlight les périphériques peuvent ,

jusqu'à un certain point , être partagés entre les assignements du pilote GoFlight et les assignements de FSUIPC . (quoiqu'il soit admis que cela produise des complications avec les affichages et les indicateurs) . la situation de VRInsight est assez différente . Il n'y a pas de chemin facile pour un programme de niveau-utilisateur tel que FS+FSUIPC de sorte à partager l'utilisation du même port COM avec le pilote VRInsight (SerialPF2) . ça pourrait probablement être fait en utilisant quelque chose comme le programme Eterlogic VSPE comme un « splitter », mais ce n'est pas une solution généraliste pour les utilisateurs .Donc ça ressemble d'abord à ce qui pourrait apparaître comme un ou bien …/ou bien … : ou bien vous utilisez SerialPF2 , ou bien vous utilisez FSUIPC probablement avec un plug-in Lua pour programmer l'affichage . Ce qui signifierait que le plug-in doit faire beaucoup de travail , dont la plus grande part au-delà des moyens de la plupart des utilisateurs .

Toutefois , le VSPE Eterlogic (« Virtual Serial Port Emulator ») offre une bonne solution . Il peut fournir n'importe quel numéro de « paire » de port série virtuels : c'est-à-dire que deux « pseudo » ports COM sont liés .Par exemple , COM9 et COM10 peuvent constituer une « paire » . Tout ce qu'un programme écrit à COM9 peut être lu par un autre programme sur COM10 et réciproquement .C'est une fonctionnalité dont j'ai déjà fait la promotion avec mes liens GPSout via WideFS, pour des applications de déplacement en cartographie .

L'utilisation de paires port série virtuels permet à FSUIPC de prendre place entre le périphérique VRInsight et le pilote VRInsight (SerialFP2) . Alors FSUIPC peut détourner quelques uns ou tous les boutons et interrupteurs vers des pratiques déterminées dans les options FSUIPC, et il peut fournir des plug-ins optionnels Lua avec des opportunités pour accrocher aux périphériques à la fois les entrées depuis les interrupteurs et les sorties à l'affichage .

OK . donc , si vous êtes toujours intéressés , allons aux instructions pour compléter :

Réglages des ports série virtuels

D'abord téléchargez le programme VSPE :

http://www.eterlogic.com/Products.VSPE.html

si vous utilisez Windows 32 bits , vous pourrez utiliser la clé d'activation gratuite , qui est incluse . Pour les utilisateurs de Windows 64 bits , ils devront en commander une (25$ U.S.) .

Après l'avoir installé et enregistré (vous devrez faire un copier-coller de la longue clé d'activation !) , procédez comme suit :

1. depuis le menu **Device** , sélectionnez **Create**
2. dans le menu déroulant **Device Type** , sélectionnez **Pair** , puis appuyez sur **Next** et **Finish** .
3. répétez les étapes 1 et 2 pour le nombre de périphériques VRI que vous voulez connecter de cette manière
4. prenez note des paires créées . par exemple

COM5 <-> COM6
COM7 <-> COM8

5. Dans le fichier Menu , sélectionnez **Save As** , et sélectionnez la configuration à un endroit dont vous connaissez le nom . par exemple , dans C:\ avec un nom tel que Compairs_56_58
   Pour convenir à l'exemple de configuration que j'ai donné plus haut.
6. Maintenant fermez VSPE . Les paires seront détruites par défaut . C'est bien .
7. trouvez le raccourci vers VSPE que l'installateur a placé sur votre Bureau. Faites un click-droit dessus , sélectionnez Properties , puis à la fin du texte dans « Target » , et après avoir ajouté un espace :
   -minimize-hide_splash C:\Compairs_56_78.vspe
   Où vous placerez votre propre chemin (path) et votre nom de fichier de configuration à la place de « C:\ComPairs_56_78"
8. Vous avez maintenant le choix . Vous pouvez faire que ce programme démarre lorsque Windows démarre – faites glisser (drag) le raccourci ou sa copie dans le répertoire **Startup** de Windows .C'est ce que je ferais . L'existence de tous ces ports supplémentaires COM ne cause aucune nuisance lorsque vous ne les utilisez pas, et vous serez ennuyé si vous oubliez de démarrer le programme avant de lancer FS .

Notez que vous ne devez pas le démarrer en utilisant un paramètre **Run** dans la section [**Programs**] de FSUIPC4.INI .Ce sera trop tard pour que FSUIPC ouvre un bout du lien pour que Serial FP2 s'y connecte .

Configurer FSUIPC4 pour maintenir les périphériques VRI .

Maintenant , nous devons éditer le fichier **FSUIPC4.INI** . Trouvez-le dans le répertoire Modules de FS . – si Windows est réglé de sorte à cacher des types de fichier connus , ça ressemblera juste à « FSUIPC4 » avec un fichier de type « Configuration Settings » .Chargez-le en mémoire dans un éditeur de texte comme Notepad – n'utilisez pas Wordpad ou un processeur de mots !

Ajoutez une section entièrement nouvelle :

[VRInsight]
1=<device>, <driver>
2=<device>, <driver>

Où les entrées <device> et <driver> sont des noms de port série .Vous avez besoin d'une ligne pour chaque périphérique VRI . L'ordre n'a pas d'importance . L'entrée <device> donne le nom de porte sérié réel du périphérique et l'entrée <driver> le nom virtuel de port série .

Vous pouvez assigner n'importe quelle paire virtuelle à n'importe quel périphérique , mais juste une seule paire à un seul périphérique . Alors , pour chaque périphérique , vous entrerez un des noms de port de la paire en tant que <driver> . L'autre nom de la paire sera utilisé par SerialPF2 – vous ne devriez pas avoir besoin de vous en préoccuper si SerialPF2 est réglé sur « Auto » , ainsi que vous le trouverez .

Exemple : supposez que j'ai un périphérique VRI sur COM3 et un autre sur COM9 . Avec mes deux paires ainsi réglées dans l'exemple de la page précédente , je pourrais avoir :

1=COM3,COM6
2=COM9,COM8

Alors SerialPF2 connecterait à la première via COM5 et à la seconde via COM7 . Voici les connexions qui seront créées :

SerialFP2 ← →COM5 ← →COM6 ← →FSUIPC ← →COM3 ← →VRI Device1

SerialFP2 ← →COM7 ← →COM8 ← →FSUIPC ← →COM9 ← →VRI Device2

Notez que si vous n'avez pas besoin que SerialPF2 pilote votre périphérique , s'il a seulement des boutons et des interrupteurs que vous vouliez assigner dans FSUIPC , ou que vous vouliez le piloter avec un plug-in Lua à la place de SerialPF2 , alors vous n'avez pas besoin d'avoir une « paire » pour cela et vous omettrez le second port dans les paramètres [VRInsight] . je ne pense pas que ça soit bon à appliquer très souvent .

Lancer SerialFP2

Pendant que vous éditez le fichier FSUIPC4.INI , vous devriez considérer la façon dont vous allez lancer SerialFP2 . Il ne doit pas être lancé avant que FSUIPC n'ait empoigné (grabbed) le port réel du périphérique , ou bien il l'obtiendra et empêchera l'accès à FSUIPC . Le lancer manuellement après avoir démarré FS est maladroit pour des raisons évidentes .

La meilleure façon est de le lancer à partir de FSUIPC . Pour ce faire , vous devrez l'ajouter à la section [Programs] du fichier .INI (ajoutez la section également si vous n'en avez pas) . Par exemple , pour deux périphériques , je pourrais avoir ceci :

[Programs]
Run1=READY,CLOSE,d: \VRInsight\SerialFP2\SerialFP2.axe
Run2=READY,CLOSE,d: \VRInsight\SerialFP2\SerialFP2.axe

Pour 2 périphériques , vous aurez besoin de 2 copies de lancement de Serial FP2 , et ainsi de suite . En posant « READY » ici , je stoppe son lancement avant que FSUIPC n'ait obtenu le port . CLOSE demande simplement à FSUIPC de le fermer lorsque FS ferme .

Une dernière chose . Tant que vous êtes sûr d'avoir les choses dans le bon ordre , vous pouvez vouloir mettre en service quelques Logging (enregistrements) spéciaux dans FSUIPC qui montreront ce qui arrive dans la chaîne : SerialFP2—FSUIPC4 – périphérique VRI .Ajoutez les

lignes suivantes à la section [General] du fichier INI pour un enregistrement (log) de toutes les entrées-sorties de toutes les parties :

```
Debug=Please
LogExtras=4
```

Si vous avez déjà une ligne « LogExtras » , changez la simplement . Vous pouvez aussi régler et changer le nombre dans l'onglet Logging de FSUIPC une fois que la ligne « Debug=Please » est là[Notez que le nombre LogExtras peut être « x4 » - hexadécimal4 , la même valeur , donc ne vous en souciez pas .]

OK . Maintenant , vous devriez être prets . Assurez-vous que vos périphériques VRI soient sur on , puis lancez Fs .

Si tout va bien , vos périphériques VRI devraient s'initialiser et commencer à travailler normalement . Le fichier Log FSUIPC4 affichera , après la phase d'initialisation , des entrées comme ceci :

```
VRI  port 1  « COM5 »  opened
VRI  driver  port  1  « COM2 »  also  opened
```

Pour chaque paire listée dans la section [VRInsight] du fichier FSUIPC4.INI , et comme chaque périphérique est vu par le pilote serialFP2 (quoique probablement mélangé (mingled) puisqu'ils sont tous multi-threading) :

```
VRI COM2 → CMDRST [ from VRI driver ]
VRI COM5 ← CMDRST [ to Device ]
VRI COM2 → CMDCON [ from VRI driver ]
VRI COM5 ← CMDCON [ to Device ]
VRI COM5 → CMDCON [ from Device ]
VRI COM2 ← CMDCON [ to VRI Driver ]
VRI COM5 → APLMAST+ [ from Device ]
VRI COM2 ← APLMAST+ [ to VRI driver ]
VRI COM2 → CMDFUN [ from VRI driver ]
VRI COM5 ← CMDFUN [ to Device ]
VRI COM5 → CMDFMER [ from Device ]
VRI FMER (“MCP Combi “ )  detected on port COM5
VRI COM2 ← CMDFMER [ to VRI Driver ]
```

Notez que FSUIPC4 reconnaît ici FMER comme étant le Combi MCP .

Si le pilote SerrialFP2 ne trouve pas le périphérique , il peut avoir besoin d'aide . Essayez de le régler sur « AUTO » et réessayez .Une fois que vous l'avez en train de travailler , ça devrait être mieux la prochaine fois .

Programmer des boutons , des interrupteurs et des poignées(pommeaux) (knobs)

Une fois que vous avez atteint ce niveau (stage) vous devriez trouver que vous pouvez détecter et programmer la plupart des boutons et interrupteurs de VRInsight avec l'onglet de FSUIPC Buttons et Switches . Ils auront des numéros de joystick 256 et au-dessus . Quelques commutateurs ressembleront à 4 boutons – fast et slow dans chaque direction . mais quelques uns n'ont pas le mode fast .

L'onglet Buttons de FSUIPC réagit seulement aux boutons lorsqu'ils passent de « on » à « off » . Pour les périphériques VRI ça signifie généralement deux pressions sur les boutons – à la différence des boutons normaux de joystick, les maintenir appuyé ne donne rien d'utile . Il n'y a pas d'indication de ceci disponible . Vous pressez et puis vous relâchez pour une indication , puis vous faites la même chose pour la suivante . Chaque fois que vous faites ceci , ça change alternativement l'état du bouton de « on » à « off » et réciproquement . Si vous voulez qu'un bouton fasse quelque chose chaque fois que vous appuyez dessus , vous aurez besoin de programmer et l'appui et le relâchement. Des considérations similaires s'appliquent d'ordinaire aux commutateurs (dials) , qui donnent à voir « on » sur un click et « off » sur un autre click , etc …

A présent les boutons et commutateurs radio ne sont pas programmables sur FSUIPC . Ils semblent fonctionner suffisamment bien tels qu'ils sont . Vous pourrez passer outre (overrideable) dans les fichiers Lua plug-ins , pour ceux d'entre vous qui souhaitent entrer dans une manipulation plus avancée des périphériques . mais ils ne sont pas adéquats (suitable) pour une ré-allocation générale .

Une fois qu'un bouton , commutateur ou interrupteur est programmé dans FSUIPC , il est caché de SerialFP2 et , en fait , du log . .

Quoi d'autres ?
Quoi au sujet de l'affichage

Bonnes questions .

Tout ce que peut faire SerialFP2 avec un périphérique peut également être fait avec un plug-in Lua , en utilisant les nouvelles fonctionnalités offertes par la nouvelle librairie « **com** » , et , avec l'aide de quelques paramètres supplémentaires qui vont dans le fichier FSUIPC4.INI , ça peut fonctionner avec Serial FP2 prenant aussi sa part , si vous ne tenez pas à tout reprogrammer vous-même .

Le package Lua fourni avec FSUIPC contient plein de détails à la fois sur le côté programmation de Lua et sur la façon dont le fichier FSUIPC.INI peut être édité pour faire en sorte que tout ceci fonctionne sans heurt et automatiquement . Deux exemples relativement simples sont inclus et expliqués :

- l'un pour permettre à l'affichage et à l'ajustement du Combi Speed du MCP de fonctionner correctement , tant en mode Mach qu'en mode IAS et
- l'autre pour échanger l'utilisation des Pouces pour le réglage de l'altimètre BARO sur le tableau M contre des millibars (ou hectoPascals si vous préférez)

Si vous possédez le MCP-Combi ou le Tableau de bord-M , vous pouvez vouloir essayer l'un de ces exemples maintenant . Des instructions sont incluses dans le package Lua ZIP .