

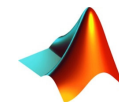
Cours MATLAB et Simulink appliqué au traitement vidéo sur DSP



Daniel MARTINS
Ingénieur d'applications
01 41 14 88 40
daniel.martins@mathworks.fr

Agenda

- 9h00 : Introduction à MATLAB : quelques commandes à savoir
- 9h30 : Introduction à Simulink : Comment fonctionne le moteur de calcul
- 9h45 : Modélisation d'un filtre de Sobel : prise en main de l'outil, gestion de la taille des données, du type.
- 10h30 : Pause
- 10h45 : Création de ses propres composants : import de code C dans Simulink
- 11h00 : Génération de code C générique
- 11h15 : Application à un DSP Texas Instruments avec création d'un projet Code Composer Studio
- 11h30 : Vérification du code objet dans Simulink : Processor In the Loop (PIL)
- 11h45 : Profile du code DSP
- 12h00 : Fin



Introduction à MATLAB

3

MATLAB Desktop

The screenshot shows the MATLAB Desktop environment with several components labeled:

- Répertoire courant**: Points to the Current Directory browser on the left side of the interface.
- Editeur de texte**: Points to the central code editor window displaying MATLAB code.
- Espace de travail**: Points to the Workspace browser on the right side, showing variables and their values.
- Fenêtre de commandes**: Points to the Command Window at the bottom, which contains the MATLAB prompt and command history.
- Historique des commandes**: Points to the Command History window, which lists previously entered commands.

The code editor shows the following MATLAB code:

```

172 % -----
173 function initialise_gui(fig_handle, handles, isreset)
174 % If the metricdata field is present and the reset flag is false, it means
175 % we are we are just re-initializing a GUI by calling it from the cmd line
176 % while it is up. So, bail out as we dont want to reset the data.
177 if isfield(handles, 'metricdata') && ~isreset
178     return;
179 end
180 handles.metricdata.density = 0;
    
```

4

Création de données

- Création d'un scalaire

```
>> a = 1.3
a =
    1.3000
```

Le point virgule après l'expression supprime l'affichage

```
>> a = 1.3;
```

- Création d'un vecteur :

```
>> b = [1 -2 13 41]
b =
     1     -2    13    41
>> c = 1:5
c =
     1     2     3     4     5
>> d = 4:-0.5:2.5
d =
    4.0000    3.5000    3.0000    2.5000
```

$n:inc:m$ est identique à $[n, n+inc, n+2*inc, \dots]$ et ne dépassera pas m

Création de données

- Création d'une matrice:

```
>> A = [1, 2; 3, 4]
A =
     1     2
     3     4
```

La virgule (ou espace) est un séparateur de colonnes

Le point-virgule est un séparateur de lignes

On peut concaténer plusieurs éléments:

```
>> B = [A 2*A]
B =
     1     2     2     4
     3     4     6     8
>> C = [A; 2*A]
C =
     1     2
     3     4
     2     4
     6     8
```

Création de données

- **Transposé d'une matrice**

```
>> D = B'
```

```
D =
```

```
 1  2
 3  4
 2  4
 6  8
```

- **Opération sur les matrices**

Multiplication matricielle -> *

```
>> E = B*C
```

```
E =
```

```
 35  50
 75 110
```

Multiplication élément par élément -> .*

```
>> F = C.*D
```

```
F =
```

```
 1  6
 6 16
 4 24
24 64
```

Indexation

- **Passer le numéro de ligne et le numéro de colonne en paramètres d'entrée**
- **L'indexation commence à 1, et le dernier élément est `end`**
- **Les vecteurs de nombres en ligne et/ou en colonne produisent un tableau qui est l'intersection des indices de ligne et de colonne**
- **Utiliser les deux-points pour spécifier un intervalle**
- **Les deux-points seuls (`:`) retournent tous les éléments dans cette dimension, ce qui est égal à `1:end`**

	1	2	3
1	1,1	1,2	1,3
2	2,1	2,2	2,3
3	3,1	3,2	3,3
4	4,1	4,2	4,3

Indexation

```
>> M(1, 3)
ans =
    1.3
>> M(2, :)
ans =
    2.1    2.2    2.3
>> M(3:4, 2:3)
ans =
    3.2    3.3
    4.2    4.3
>> M([1 3:4], 3)
ans =
    1.3
    3.3
    4.3
>> M(1, end)
ans =
    1.3
```

	1	2	3
1	1,1	1,2	1,3
2	2,1	2,2	2,3
3	3,1	3,2	3,3
4	4,1	4,2	4,3

↑
M

Quelques commandes utiles

- **whos** Affiche les variables de l'espace de travail avec leur type et leur dimension
- **size** Détermine les dimensions d'une matrice. Accepte 2 arguments, le premier étant la matrice et le deuxième étant la dimension que l'on cherche (1 pour le nombre de lignes et 2 pour le nombre de colonnes)


```
>> A = [1 2 3 4; 5 6 7 8];
>> size(A,1)
ans =
    2
>> size(A,2)
ans =
    4
```

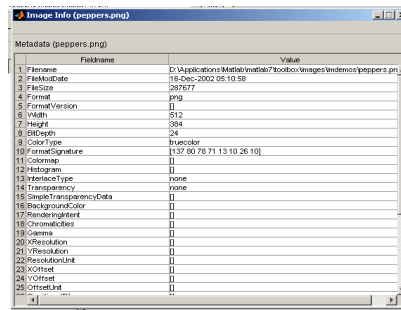
Lecture et écriture d'images

La lecture d'images se fait par `imread`.

L'écriture par `imwrite`.

`imfinfo` permet d'obtenir des informations sur l'image contenue dans un fichier.

`imageinfo` permet de récupérer les mêmes informations que la fonction `imfinfo` mais sous une interface graphique.



Les formats supportés : bmp, hdf, jpeg, pcx, tiff, xwd.

Représentation Graphique de l'importation d'une image



file.fmt

`imfinfo('file.fmt')`

Colormap?

oui

non

`x=imread('file.fmt');`

`[x,map]=imread('file.fmt');`

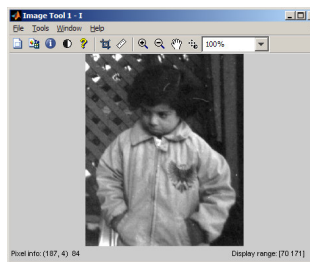
Image Type	Indexed
Double Data	$[MXN]$
Uint8 Data	$[MXN]$

Image Type	Intensity	Binary	RGB
Double Data	$\begin{bmatrix} 1 & 0.5 & 0.2 \\ 0 & 0.1 & 0.9 \\ 0 & 0.7 & 0.8 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0.5 & 0.2 \\ 0.9 & 0.2 & 0.3 \end{bmatrix}$
Uint8 Data	$\begin{bmatrix} 1 & 45 & 220 \\ 100 & 78 & 110 \\ 200 & 7 & 98 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 255 & 130 & 51 \\ 10 & 80 & 255 \\ 109 & 220 & 230 \end{bmatrix}$

Affichage d'images

- `imtool` - Affiche l'image dans le visualiseur d'images.
- `imshow` - Affiche l'image.
- `image` - Créé et affiche un objet image (MATLAB).
- `imagesc` - Met à l'échelle et affiche les images (MATLAB).
- `colorbar` - Affiche la barre des couleurs (MATLAB).
- `colormap` - Assigne une carte des couleurs à l'image (MATLAB).
- `truesize` - Ajuste l'affichage à la taille de l'image.

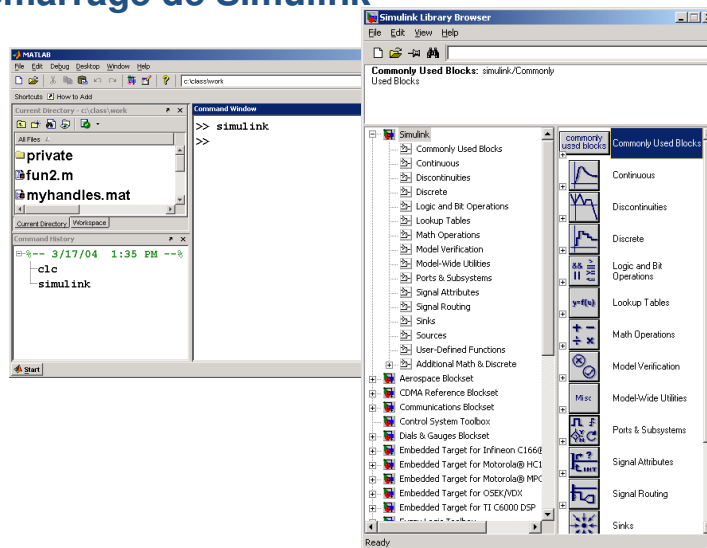
```
>> I = imread('pout.tif');  
>> imtool(I)
```



Introduction à Simulink

Création d'un modèle Simulink

Démarrage de Simulink



Explorateur de bibliothèques Simulink

Simulation Library Browser

File Edit View Help

Bus Creator: This block creates a bus signal from its inputs.

Recherche de bloc

Description du bloc

Bibliothèque

Sous-bibliothèque

Ready

17

Création d'un nouveau schéma-bloc

Simulation Library Browser

File Edit View Help

New Model Ctrl+N

Open... Ctrl+O

Close

Library

Preferences...

Barre d'outils Simulink

Icône Nouveau modèle

Simulation Library Browser

File Edit View Simulation Format Tools Help

Go To Parent

- ✓ Toolbar
- Status Bar
- Model Browser Options
- Block Data Tips Options
- System Requirements
- Library Browser
- Model Explorer
- Zoom In
- Zoom Out
- Fit System To View
- Normal (100%)
- Remove Highlighting
- Highlight...

Show or hide the toolbar

100%

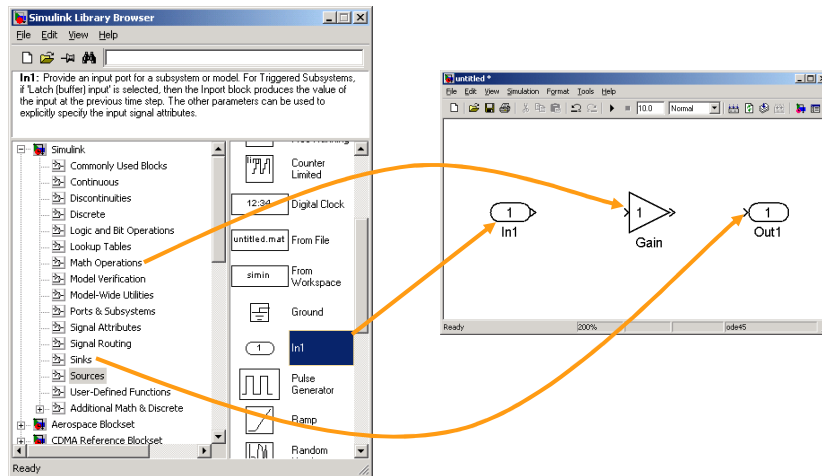
ode45

Barre d'état

18

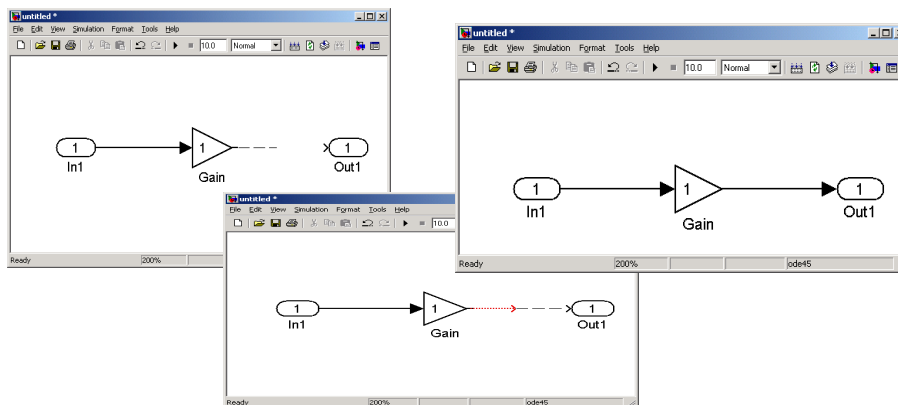
Création d'un nouveau schéma : ajout de blocs

Pour ajouter un bloc à un modèle, faites-le glisser dans le schéma-bloc depuis l'explorateur de bibliothèques. Les blocs sont donnés en liste alphabétique dans chaque bibliothèque.



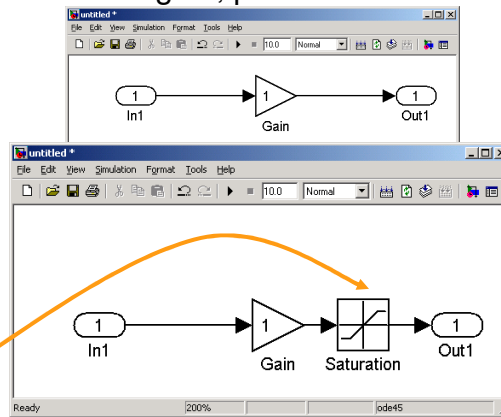
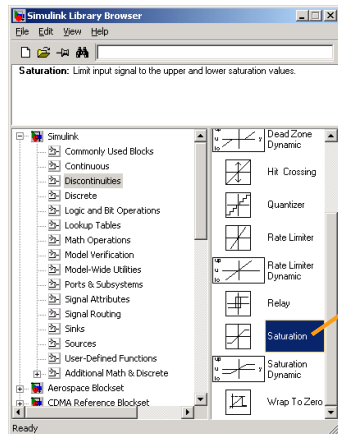
Création d'un nouveau schéma : ajout de signaux

Pour connecter deux blocs avec un signal, glissez depuis le port de sortie du bloc source vers le port d'entrée du bloc cible.



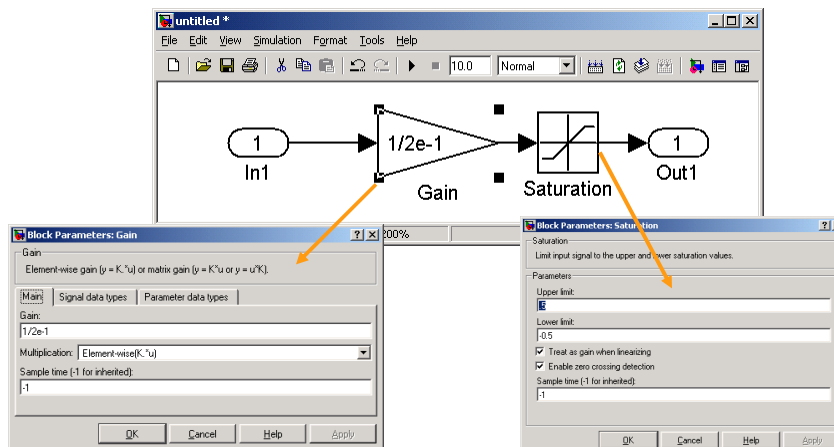
Édition du schéma-bloc : insertion de blocs

Pour insérer un bloc au milieu d'un signal, placez le bloc sur la ligne de signal.



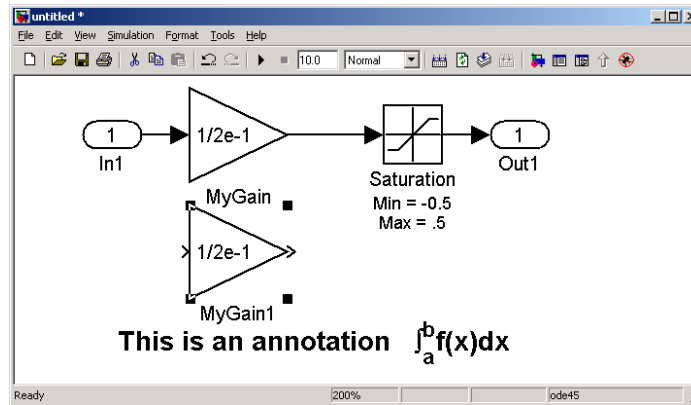
Définition des paramètres d'un bloc

- Pour ouvrir le dialogue des paramètres de bloc, double-cliquez sur le bloc
- Entrez une valeur correcte ou une sélection dans chaque champ.



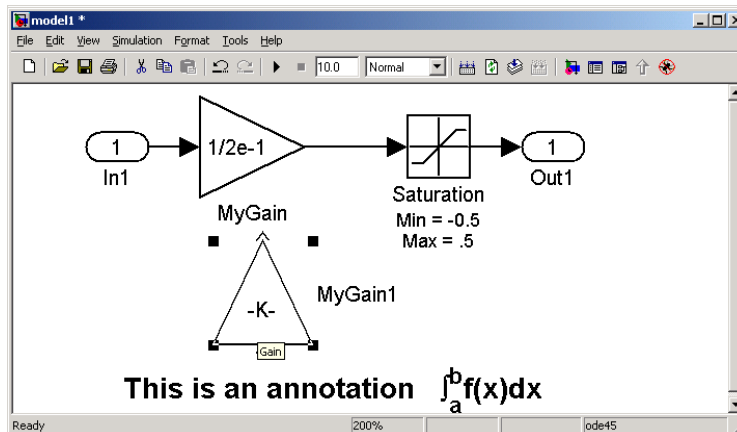
Copie de blocs

Pour copier un bloc, cliquez sur le bloc en utilisant le bouton droit de la souris et faites-le glisser.



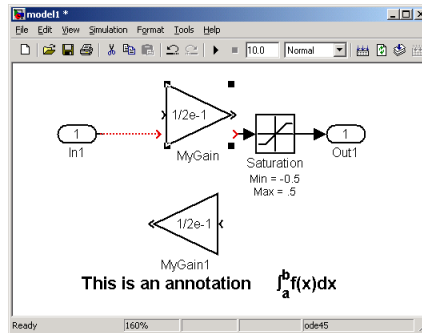
Réorientation des blocs

- Pour retourner un bloc de gauche à droite, sélectionnez Flip block dans le menu Format après avoir sélectionné le bloc.
- Pour faire pivoter un bloc dans le sens des aiguilles d'une montre 90°, sélectionnez Rotate block dans le menu Format après avoir sélectionné le bloc.



Déconnexion des blocs

Pour déconnecter un bloc des signaux qui y sont attachés, cliquez sur le bloc avec le bouton droit de la souris et éloignez-le tout en pressant la touche Maj.

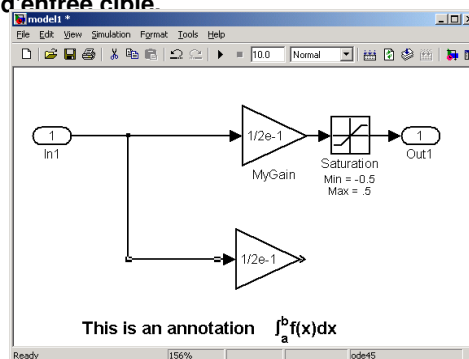


Maj + clic droit

25

Raccordement de signaux

Pour ajouter un embranchement de signal, cliquez bouton droit sur le signal et glissez depuis l'endroit voulu de la ligne de signal jusqu'au port d'entrée cible.

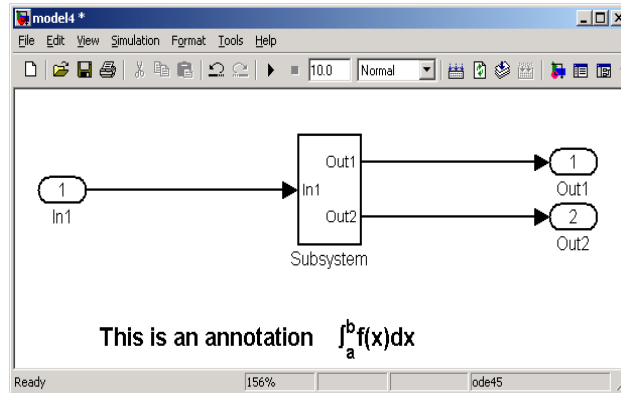


>> model1

26

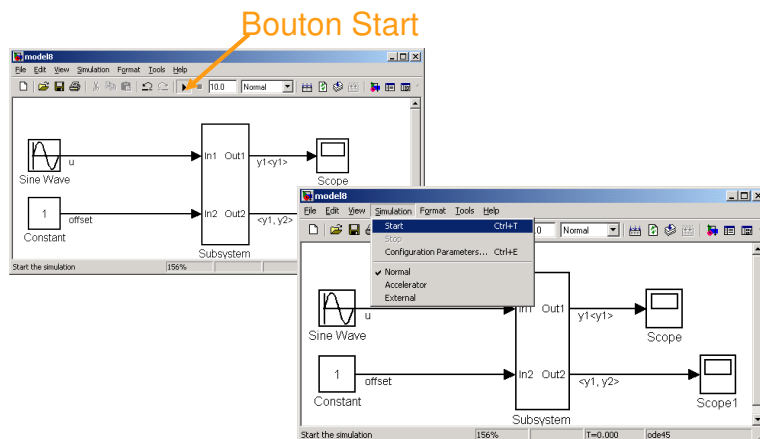
Création de sous-systèmes

Pour grouper des blocs en un sous-système, sélectionnez les blocs dont vous voulez qu'ils soient contenus dans le sous-système, puis sélectionnez Create subsystem dans le menu Edit.



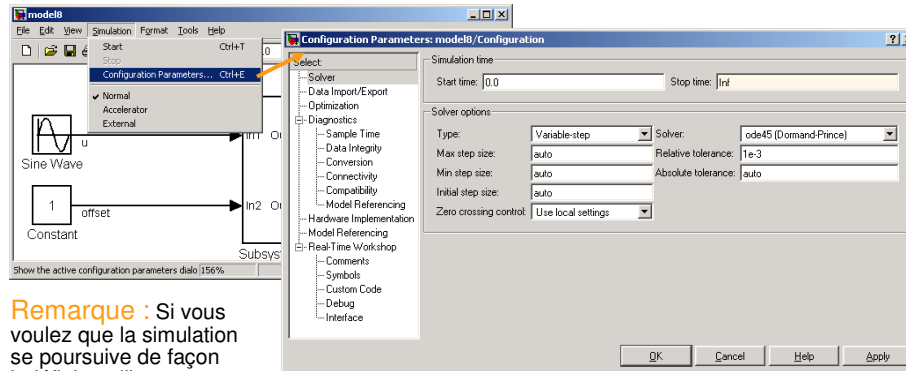
Exécution du modèle avec les paramètres par défaut

Pour simuler le modèle, pressez le bouton  ou sélectionnez l'option Start dans le menu Simulation.



Réglage du temps de simulation

- Ouvrez les paramètres de configuration en sélectionnant Configuration Parameters dans le menu Simulation.
- Réglez les temps de simulation en utilisant les champs Start time et Stop time dans la section Solver de la boîte de dialogue Configuration Parameters.



Remarque : Si vous voulez que la simulation se poursuive de façon indéfinie, utilisez `inf` comme `Stop time`.

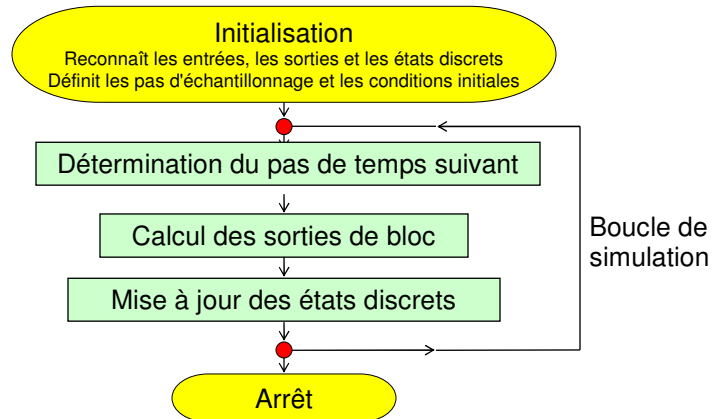
Deuxième modèle

29

Le moteur de calcul de Simulink

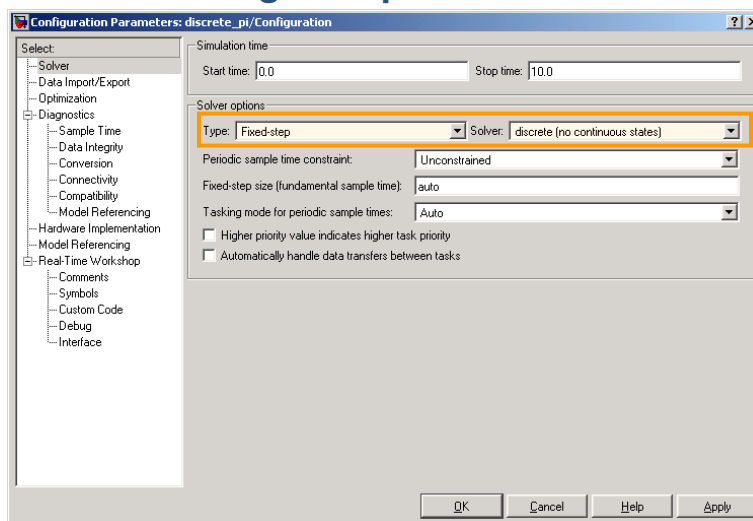
30

Simulation de systèmes discrets

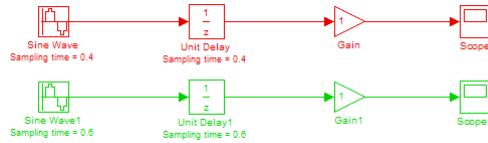


Systèmes discrets

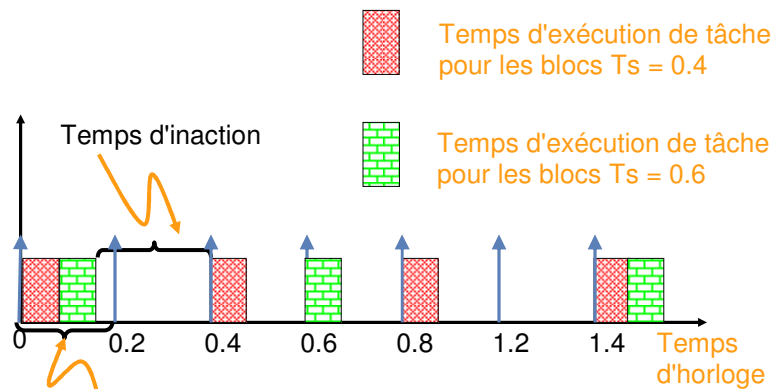
Choix d'un solveur pour un modèle à période d'échantillonnage unique



Solveurs discrets à périodes multiples



Périodes multiples en monotâche (suite)

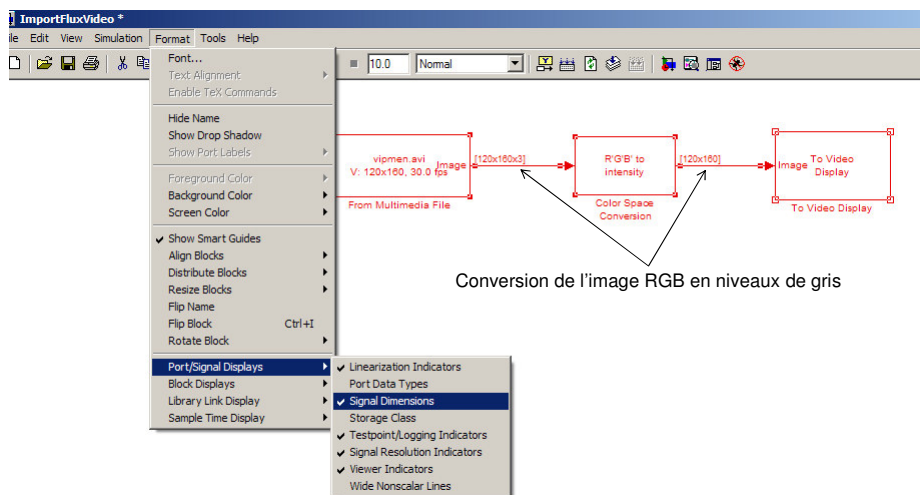


Pas d'échantillonnage fondamental

- doit être plus grand que la somme des durées d'exécution des tâches rapides et lentes
- doit être un diviseur commun de toutes les périodes d'échantillonnage

Détection de contour par filtrage de Sobel

Import d'un flux video



Détection de contour par filtrage de Sobel

Annotations in the image:

- Pointing to the top FIR filter: Filtrage sur les lignes
- Pointing to the bottom FIR filter: Filtrage sur les colonnes en utilisant la matrice transposée

Parameter windows for the FIR filters show:

- Top FIR Filter: Coefficients: $[1 \ 2 \ 1; 0 \ 0 \ 0]$
- Bottom FIR Filter: Coefficients: $[1 \ 2 \ 1; 0 \ 0; -1 \ -2 \ -1]$

Page number: 37

Détection de contour par filtrage de Sobel

Annotation in the image:

- Pointing to the 'Compare To Constant' block: La sortie du comparateur doit être booléenne

Parameter window for the 'Compare To Constant' block shows:

- Operator: \geq
- Constant value: 0.5
- Output data type mode: boolean

Page number: 38

Import de code C

Import de code C

Pour importer un code C dans Simulink, il faut passer par la création d'une S-fonction. Cette création peut être automatiser apr le biais d'un script à base du legacy_code tool

Simulink supporte les pointeurs sur des entiers, des flottants mais pas des caractères. Pour importer une fonction C dans Simulink, celle-ci doit avoir un prototype avec des arguments valides constitués de pointeurs ou valeurs sur des types entiers ou flottants. Au besoin il faudra créer une fonction Wrapper acceptant des arguments valides qui appellera la fonction souhaitée utilisant des arguments non valides

Exemple de fonction code C

La fonction GradientImage utilise le type non valide PTR_PICTURE pour lire et modifier des images

```
void GradientImage( PTR_PICTURE pImgSource, PTR_PICTURE pImgGraduee ){
    register PTR_PIXEL pGraduee;
    register PTR_PIXEL pSource0;
    ...}
```

Il faut créer une fonction wrapper qui n'accepte que des pointeurs et/ou valeurs sur des types entiers ou flottants

```
void wrapperGradient(int *imgSource, int *imgLisee, int iLargeur, int iHauteur)
{
    PICTURE PictureSource, PictureLisee;

    PictureSource.iHauteur = iHauteur;
    PictureSource.iLargeur = iLargeur;
    PictureSource.pPixels = imgSource;
    PictureLisee.iHauteur = iHauteur;
    PictureLisee.iLargeur = iLargeur;
    PictureLisee.pPixels = imgLisee;
    GradientImage(&PictureSource, &PictureLisee);
}
```

41

Le legacy_code tool

Le script commence par la création d'un objet

```
def = legacy_code('initialize');
```

Puis d'un nom pour la S-fonction

```
def.SFunctionName = 'sfun_Gradient';
```

Puis du prototype de la fonction valide

```
void wrapperGradient(int *imgSource,
int *imgLisee, int iLargeur, int iHauteur)

def.OutputFcnSpec = 'void wrapperGradient(uint8 u1[][],
uint8 y1[size(u1,1)][size(u1,2)], int16 size(u1,1), int16 size(u1,2))';
```

u1, u2, ...ui est utilisé pour les entrées 1, 2, ...i

y1, y2, ...yj est utilisé pour les sorties 1, 2, ...j

p1, p2, ...pk est utilisé pour les paramètres 1, 2, ...k

u1 est utilisé pour une valeur

u1[] est utilisé pour un pointeur sur un vecteur

u1[][] est utilisé pour un pointeur sur une matrice

y1[size(u1,1)][size(u1,2)] : pointeur sur une matrice de même taille que u1

size(u1,1) pour iLargeur

size(u1,2) pour iHauteur

42

Le legacy_code tool (suite)

On termine le script par:

Les fichiers .C et .H nécessaires

```
def.HeaderFiles = {'Algo.h'};  
def.SourceFiles = {'Roberts.c', 'WrapRoberts.c'};
```

La création de la S-fonction proprement dit

```
legacy_code('sfcn_cmex_generate', def);
```

La compilation de la S-fonction

```
legacy_code('compile', def);
```

La création d'un bloc pour Simulink

```
legacy_code('slblock_generate', def);
```

La création d'un fichier utile pour la génération de code

```
legacy_code('sfcn_tlc_generate', def);
```

Import de code MATLAB

Ecrire une fonction Embedded MATLAB

The screenshot shows the MATLAB/Simulink environment. On the left, the Embedded MATLAB Function block is selected in the library browser. An orange arrow points from this block to the Embedded MATLAB Editor window. The editor window shows the following code:

```

1 function y = fcn(u)
2 % This block supports an embeddable subset of the MATLAB language.
3 % See the help menu for details.
4
5 y = u;
    
```

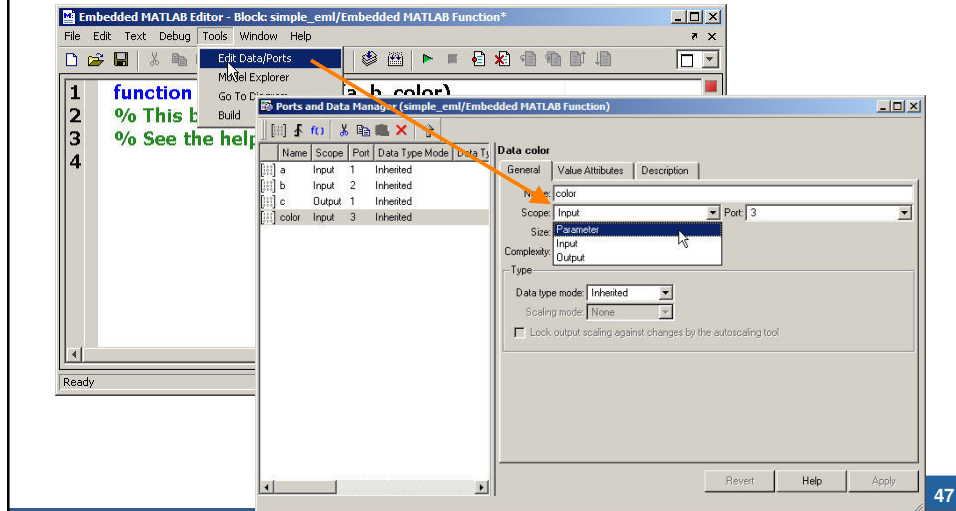
Definir les entrées et sorties du bloc

The screenshot shows the MATLAB/Simulink environment. A Simulink model contains a block named 'pythagoras' with two input ports labeled 'a' and 'b', and one output port labeled 'c'. The Embedded MATLAB Editor window is open, showing the following code:

```

1 function c = pythagoras(a,b)
2 % This block supports an embeddable subset of the MATLAB
3 % See the help menu for details.
4
    
```

Definir des paramètres



47

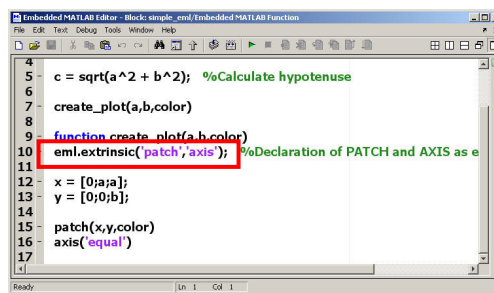
Utilisation des fonctions de la librairie Run-Time

- **Sous-ensemble des fonctions MATLAB**
- **Même nom, arguments, et fonctionnalités qu'avec MATLAB**
- **Generation de code C avec Real-Time Workshop**

48

Appel de fonctions MATLAB non run-time

- Executées par MATLAB pendant la simulation
 - La simulation sera plus lente
- Avant l'appel de telles fonctions, les déclarer en extrinsic
 - Extrinsic declaration → `eml.extrinsic`
- Aucun code de ces fonctions ne sera généré



```
4  
5 - c = sqrt(a^2 + b^2); %Calculate hypotenuse  
6  
7 - create_plot(a,b,color)  
8  
9 - function create_plot(a,b,color)  
10 - eml.extrinsic('patch','axis'); %Declaration of PATCH and AXIS as e  
11  
12 - x = [0;a];  
13 - y = [0;b];  
14  
15 - patch(x,y,color)  
16 - axis('equal')  
17
```

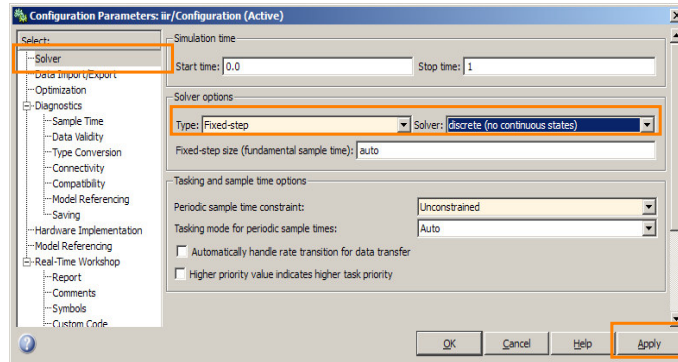
49

Génération de code C générique

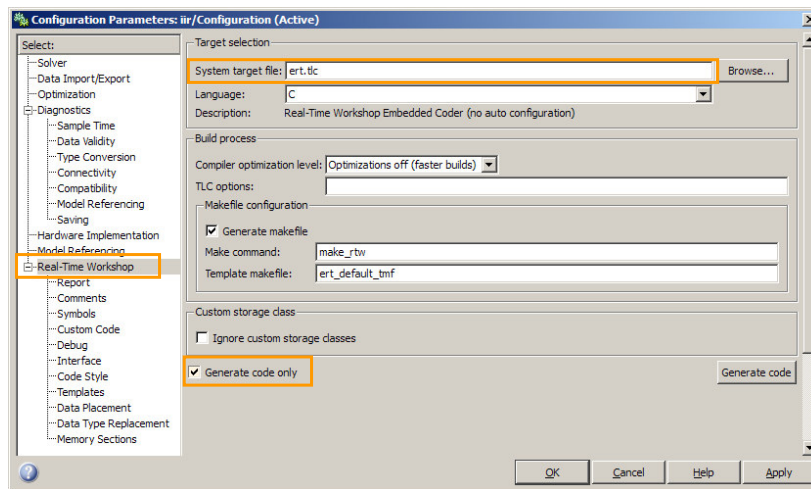
50

Pré-requis pour la génération de code

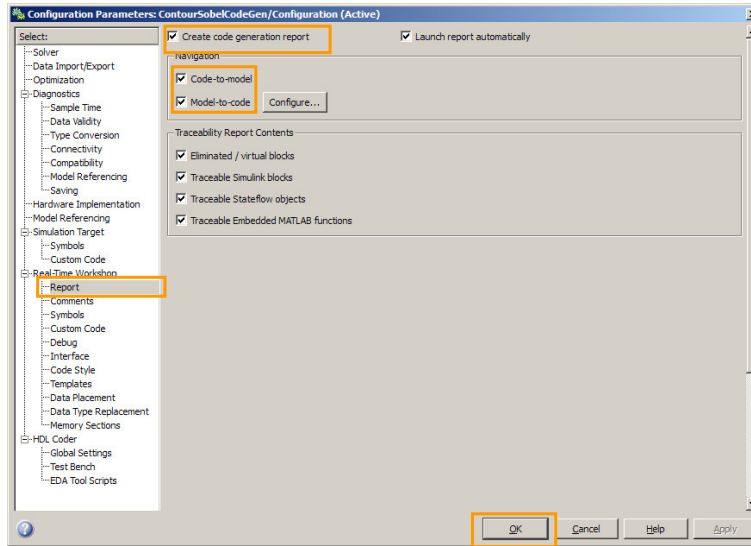
Selectionner Simulation → Configuration Parameters (Ctrl+E)



Configurer la cible

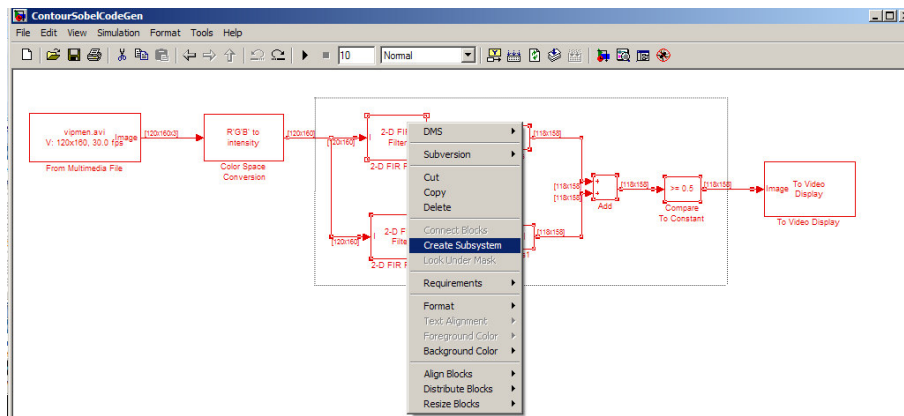


Générer un rapport de génération de code



Créer un sous-système

Selectionner les blocs dont on veut générer le code
Click bouton droit sur un des blocs sélectionnés



Générer le code

Click bouton droit sur le sous-système
Puis Build sur la 2ème interface

The screenshot shows a Simulink model window titled 'ContourSobelCodeGen'. A context menu is open over a subsystem, with the path 'Real-Time Workshop > Build Subsystem...' selected. In the foreground, a dialog box titled 'Build code for SubsystemNonAlgo' is displayed, showing a table for tunable parameters and a 'Build' button highlighted with a yellow box.

Traçabilité bi-directionnelle

- Du code au modèle

The screenshot illustrates bi-directional traceability. The 'Configuration Parameters: iir/Configuration (Active)' dialog has 'Code-to-model' checked. The 'Real-Time Workshop Report' shows a list of generated source files including 'iir_data.c', 'iir_data.h', 'iir_private.h', 'iir_types.h', and 'rtwtypes.h'. A code snippet is shown with a line 'UnitDelay: <S22>/Unit Delay' circled in red. An arrow points from this line to the 'Unit Delay' block in the Simulink model 'iir/iir_filter'.

Traçabilité bi-directionnelle

- Du modèle au code

The screenshot shows the MATLAB/Simulink environment. On the left, the 'Configuration Parameters' dialog is open, with 'Code-to-model' and 'Model-to-code' checked. The 'Real-Time Workshop Report' window is open, displaying the generated C code for the 'iir_step' function. The code includes comments and function definitions. A context menu is visible over the report, with 'Navigate to Code...' selected, indicating the bidirectional traceability from the model to the code.

Les fichiers générés don't un main "exemple"

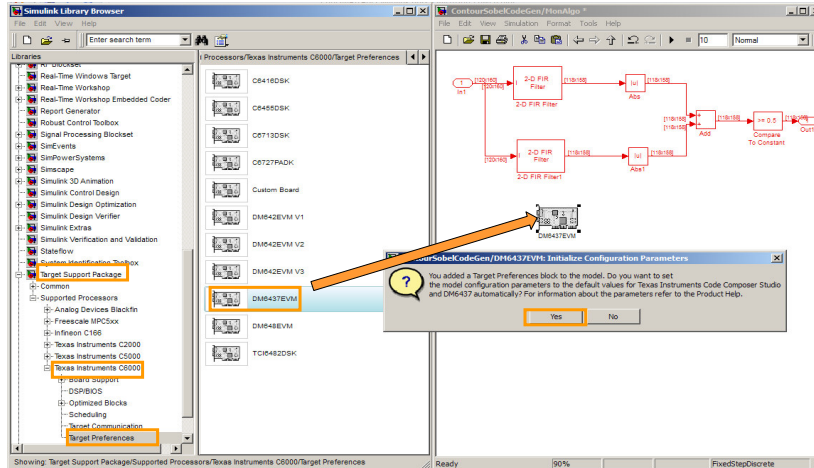
The screenshot displays two snippets of generated C code. The first snippet, titled 'ert_main.c', shows the main function which calls 'iir_initialize()', 'rt_OneStep()', and 'iir_terminate()'. The second snippet shows the 'void rt_OneStep(void)' function, which calls 'iir_step()'. A third snippet shows the 'void iir_step(void)' function, which performs the actual signal processing. The code is annotated with boxes and arrows to show the flow of execution and the relationship between the different functions.

Génération de code C pour DSP TI C6000

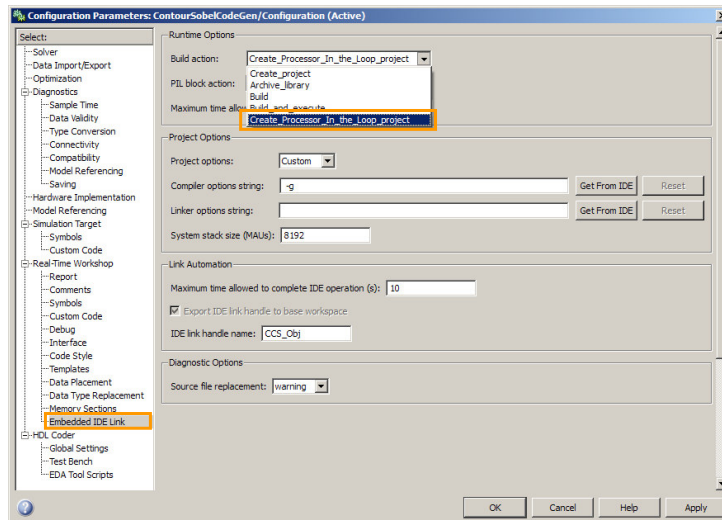
Verification de code (PIL)

Filter Algorithm PIL Verification with C6713DSK

Entrer dans le sous-système et ajouter le bloc DM6437EVM



Configurer pour Porcessor in the Loop (PIL)



Génération d'un projet CCS

Nouveau modèle à partir du sous-système

Ajouter les blocs:

- drivers video entrée/sortie
- Interleaver/deinterleaver
- Gain pour passer des types uint8 à single
- Constant pour ajouter des zeros

Configurer pour une cible DM6437

The screenshot shows a Simulink model titled 'untitled*' with various blocks including 'DM6437EVM', 'C6000', 'Gain1', 'MonAlgo', 'Constant', 'Interleave', and 'DM643x'. A 'Target Preferences' block is highlighted in the library browser. A dialog box titled 'ContourSobel/DM6437EVM: Initialize Configuration Parameters' is displayed, asking: 'You added a Target Preferences block to the model. Do you want to set the model configuration parameters to the default values for Texas Instruments Code Composer Studio and DM6437 automatically? For information about the parameters refer to the Product Help.' The 'Yes' button is highlighted.

Générer et exécuter le code sur la cible

The screenshot shows the Simulink model from the previous slide. The 'Real-Time Workshop' menu is open, showing options like 'Build Model', 'Generate S-Function...', and 'xPC Target Explorer'. A code editor window is open, displaying the generated C code for the target. The code includes headers for 'MonAlgo.h', 'Interleave.h', and 'C6000.h', and contains a main function that processes video frames. The code is as follows:

```

#include "MonAlgo.h"
#include "Interleave.h"
#include "C6000.h"
#include "videoCapture.h"
#include "videoDisplay.h"

int main()
{
    /* Initialize the video capture and display */
    videoCaptureInit();
    videoDisplayInit();

    /* Initialize the MonAlgo and Interleave blocks */
    MonAlgoInit();
    InterleaveInit();

    /* Main loop */
    while(1)
    {
        /* Capture video frame */
        videoCaptureGetFrame();

        /* Process video frame */
        MonAlgoProcess();

        /* Interleave video frame */
        InterleaveProcess();

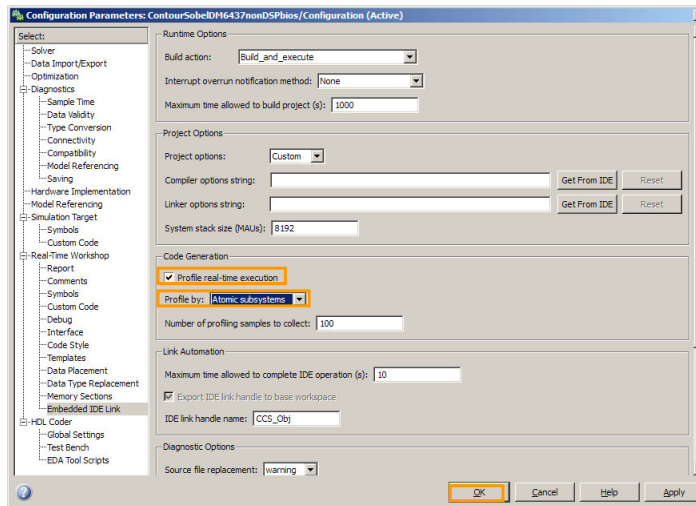
        /* Display video frame */
        videoDisplayShowFrame();
    }
}
    
```

Profile de code

Rendre le bloc MonAlgo "atomic"

The screenshot displays a Simulink model window titled 'ContourSobcDM6437nonDSPbios'. The model contains several blocks including 'Video Capture', 'DM6437EVM', 'YCbCr', 'Deinterleave', 'Gain1', 'single', 'Mon', 'Interleave', 'YCbCr', and 'Video Display'. A context menu is open over the 'Mon' block, listing options such as 'Subversion', 'Open Block In New Window', 'Cut', 'Copy', and 'Delete'. The 'Subsystem Parameters...' option is highlighted. In the foreground, the 'Function Block Parameters: MonAlgo' dialog box is open. The 'Treat as atomic unit' checkbox is checked, and an orange arrow points from this checkbox to the 'Subsystem Parameters...' option in the context menu. Other parameters in the dialog include 'Show port labels' set to 'FromPortIcon', 'Read/Write permissions' set to 'Read/Write', and 'Permit hierarchical resolution' set to 'All'.

Configurer le modèle pour le profiling (Ctrl+E)



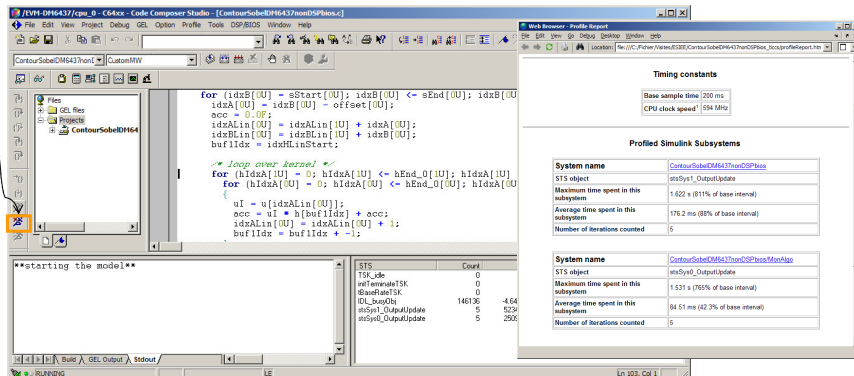
Récupérer le rapport de profile de code

Générer et exécuter le code sur cible (Ctrl+B)

Arrêter l'exécution au bout de quelques secondes

Dans MATLAB lancer la commande :

>> profile(CCS_Obj, 'execution', 'report')



Création d'une tâche DSP/Bios

The screenshot shows the Simulink Library Browser on the left, with the 'Task' block selected under 'Processors/Texas Instruments C6000/DSPBIOS'. An orange arrow points from this block to the 'DSP/Bios Task' block in the Simulink diagram. Another orange arrow points from the 'Function-Call Subsystem' block in the library to the 'Function-Call Subsystem' block in the diagram. The diagram shows a 'DSP/Bios Task' block connected to a 'Function-Call Subsystem' block with 'In1' and 'Out1' ports.

Ajout des drivers Video

Ajouter les blocs propres à la DM6437

- drivers Entrée/Sortie Video
- Entrelaceur/désentrelaceur
- La cible calcule la nouvelle image et la renvoie vers Simulink

The screenshot shows the Simulink Library Browser on the left, with the 'Video Capture' and 'Video Display' blocks selected under 'Instruments C6000/Board Support/DM6437 EVM'. Orange arrows point from these blocks to the corresponding blocks in the Simulink diagram. The diagram shows a 'Video Capture' block connected to a 'Deinterleave' block, which is connected to a 'Gain' block. The 'Gain' block is connected to a 'Mask Algo' block, which is connected to an 'Interleave' block. The 'Interleave' block is connected to a 'Video Display' block. The diagram also shows a 'DM6437 EVM' block connected to the 'Video Capture' and 'Video Display' blocks.

Générer le code au niveau du modèle

Code Composer se lance et un nouveau projet est créé, compilé, linké téléchargé sur la cible et l'exécution démarre

The screenshot displays the MATLAB/Simulink Code Composer Studio interface. The main workspace shows a Simulink model with a 'DSPBIOS' block and a 'Task' block. The 'Task' block is highlighted with a red box. The 'Build Model' menu option is selected, which is highlighted in blue. The menu path is: **Build Model** (Ctrl+B) > **Build Subsystem...** > **Generate S-Function...** > **Navigate to Code...** > **xPC Target Explorer**. The 'xPC Target Explorer' window is open, showing the 'Build Model' menu option. The 'Build Model' menu option is highlighted in blue. The 'Build Model' menu option is highlighted in blue. The 'Build Model' menu option is highlighted in blue.

```

% Abs: 'QSP>Abs' <<>
for (l = 0; l < 1044; l++) {
    MonaAlgo_B_Abs(l) = (real32_T)Eabs(MonaAlgo,
}
% Sqr: 'QSP>Sqr' <<>
for (l = 0; l < 1044; l++) {
    MonaAlgo_B_Add(l) = MonaAlgo_B_Add(l) + Mona
}
% Relations/Operators: 'QSP>Compare' Incomp
% Constant: 'QSP>Constant' <<>
    
```

Clear L2 Cache in DDR Range
Setup Cache (L1P = 32K) + (L1D = 32K) + (L2 = ALL SRAM)... [Done]
Disable EDMA events

File: C:\Inher\Work\ESSE\MonAlgo_top

Generate RTW code.

75