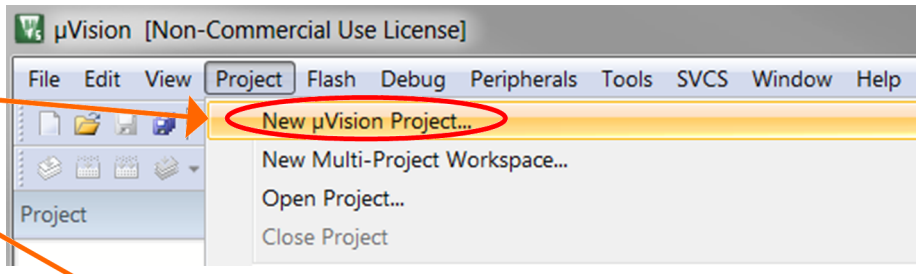
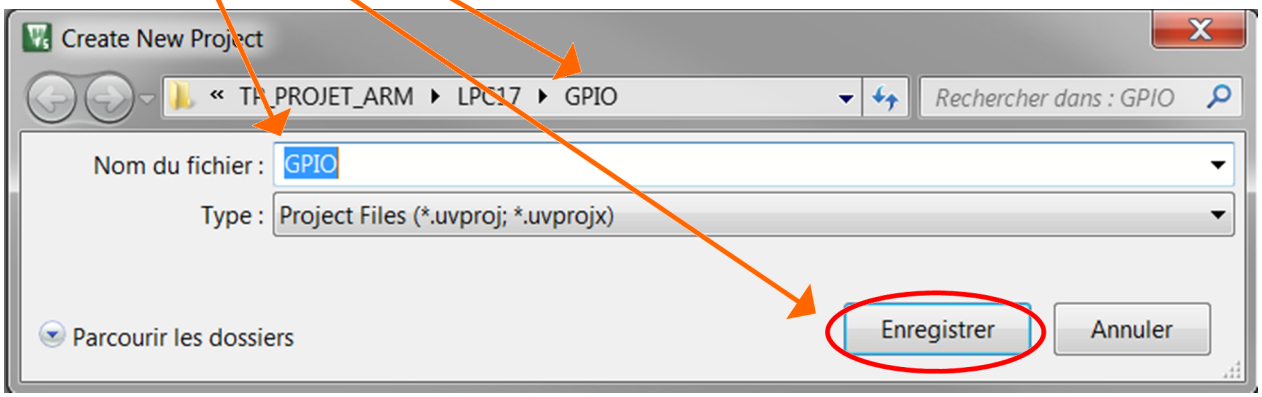
Premier projet en KEIL5 en ciblant un LPC1778 ou LPC1769

On va créer un nouveau projet :
Menu Project / New µVision Project ...

On pensera à créer un nouveau répertoire
(astuce utiliser le même nom que votre projet)



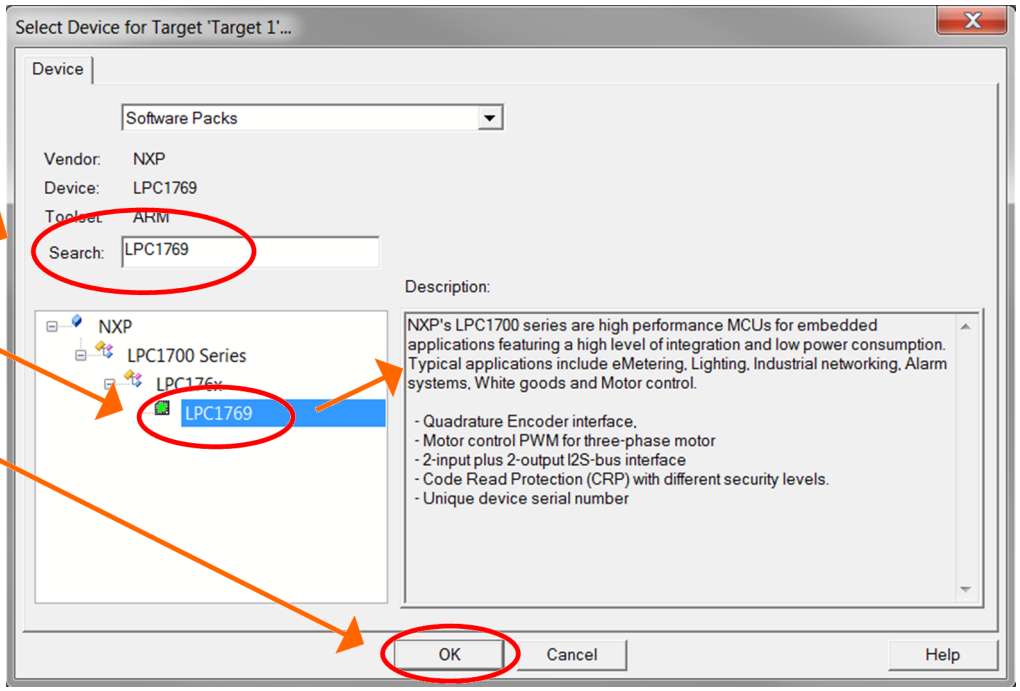


Le menu de choix de la cible apparaît :

dans le champ Search taper le nom de la cible LPC1769

Ensuite cliquer sur le processeur , ce qui fait apparaître la description

valider ce choix de processeur



ARRRRGGGGGGGGG, impossible de trouver le LPC1769 !!! ??? !!!!!

en ce cas suivre le tutoriel AJOUTER_PACKAGES.pdf
pour faire ajouter à KEIL le bon package concernant la cible choisie

puis reprendre ce tutoriel

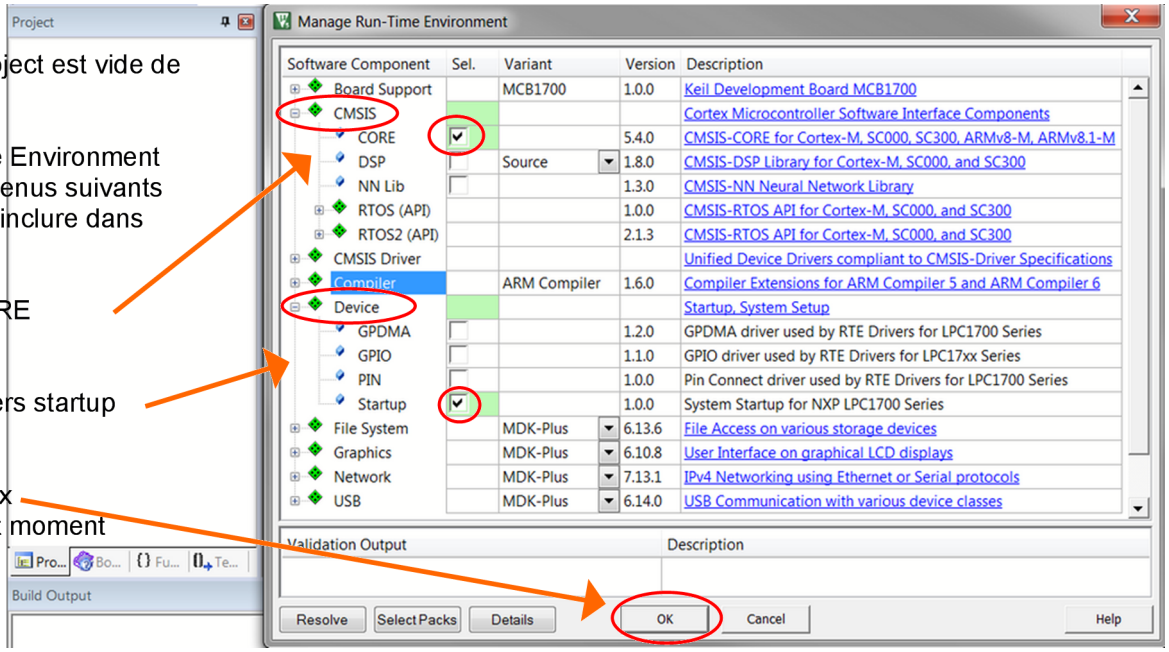
Pour le moment la fenêtre Project est vide de tout fichier

La fenêtre Manage Run-Time Environment apparaît, il faut étendre les menus suivants et selectionner les modules à inclure dans notre nouveau projet:

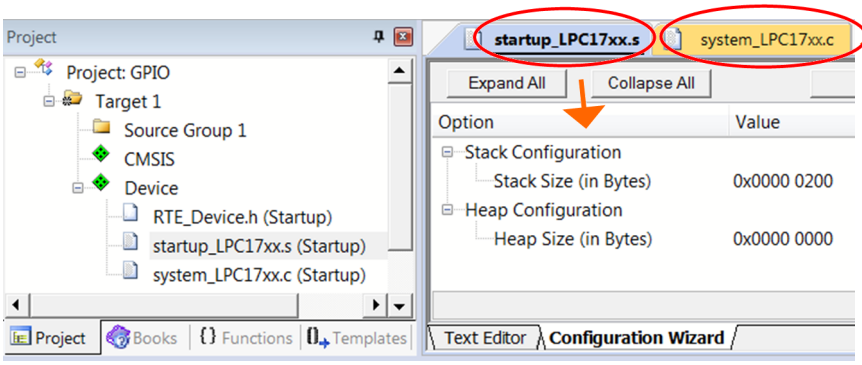
-- CMSIS pour inclure le CORE

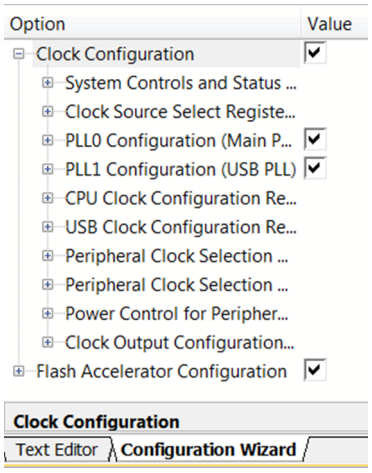
--Device pour inclure les fichiers startup

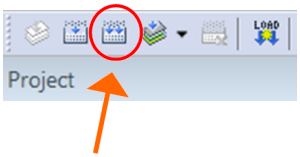
On peut alors valider ces choix que l'on pourra changer à tout moment en cliquant sur le bouton :



le projet contient maintenant cette structure de fichiers :







ce fichier Startup_LPC17xx.s configure les tailles de réservation mémoire de la pile et du tas

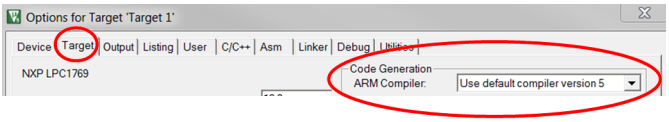
Le fichier system_LPC17xx.c gère :
-- les sources et vitesses d'horloge des divers modules du processeur (coeur, usb, périphériques)
-- les mises sous tensions des modules
-- le cadencement de la mémoire flash

Si on compile maintenant, on obtient une erreur et 25 Warning ....

```
25 warnings generated.
compiling system_LPC17xx.c...
linking...
.\Objects\gpio.axf: Error: L6218E: Undefined symbol main (referred
Not enough information to list image symbols.
Not enough information to list load addresses in the image map.
Finished: 2 information, 0 warning and 1 error messages.
".\Objects\gpio.axf" - 1 Error(s), 25 Warning(s).
Target not created.
```

```
24 warning: anonymous unions are a C11 extension [-Wc11-extensions]
1 warning: implicit conversion changes signedness: 'int' to 'unsigned int' [-Wsign-conversion]
```

Changer de chaîne de compilation, cliquez



L'erreur est logique à ce stade, puisqu'on a pas de fichier C inclus dans notre projet contenant une fonction main() ...
Les warning sont dus au choix de la chaîne de compilation
Après les changements ci-contre, on obtient le résultat suivant :

```
assembling startup_LPC17xx.s...
compiling system_LPC17xx.c...
linking...
.\Objects\gpio.axf: Error: L6218E: Undefined symbol main (referred
Not enough information to list image symbols.
Not enough information to list load addresses in the image map.
Finished: 2 information, 0 warning and 1 error messages.
".\Objects\gpio.axf" - 1 Error(s), 0 Warning(s).
Target not created.
```

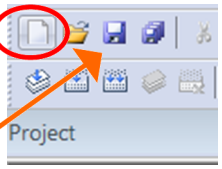
Dans l'onglet Target, changer de compilateur pour basculer sur la version 5

Ce qui est totalement délirant...c'est qu'après avoir compilé en version 5, vous pouvez revenir en version 6 et vous n'aurez plus de warning ...
En fait passer en version 5 change des options qui restent quand on retourne en version 6
voir Annexe 1 pour détails des options

Il faut maintenant résoudre l'erreur de compilation en créant une fonction int main(void)

AJOUTER NOTRE FICHIER CONTENANT une fonction main()

créer un nouveau fichier



taper le code ci contre

```
unsigned char test=0;

void init_proc(void)
{
}

int main (void)
{
init_proc();
 while(1)
  { test++;
  }
}
```

sauvegarder en gpio.c

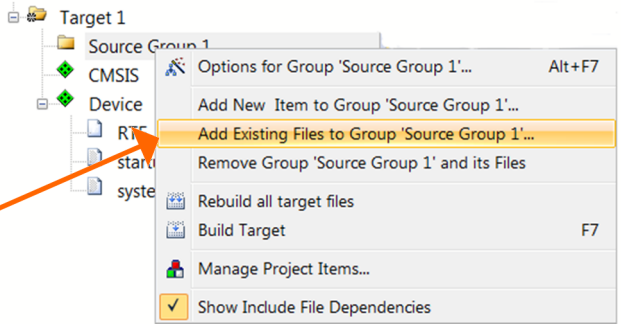
la coloration syntaxique doit apparaître :

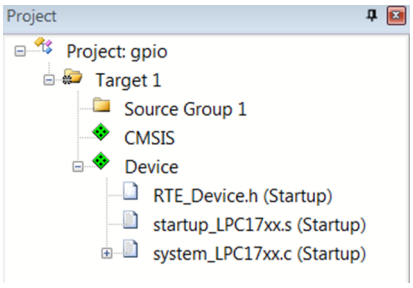
Si elle n'apparaît pas, ce n'est pas un fichier .c et vous ne le verrez pas à l'étape suivante ! il faut alors aller changer l'extension, ou sauvegarder à nouveau.

```
1  unsigned char test=0;
2
3  void init_proc(void)
4  {
5  }
6  int main (void)
7  {
8  init_proc();
9   while(1)
10    { test++;
11    }
12 }
```

AJOUTER LE FICHIER AU PROJET

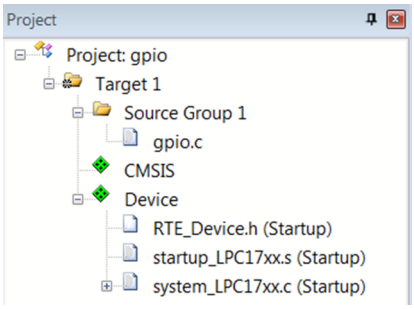
clic droit sur Source Group 1





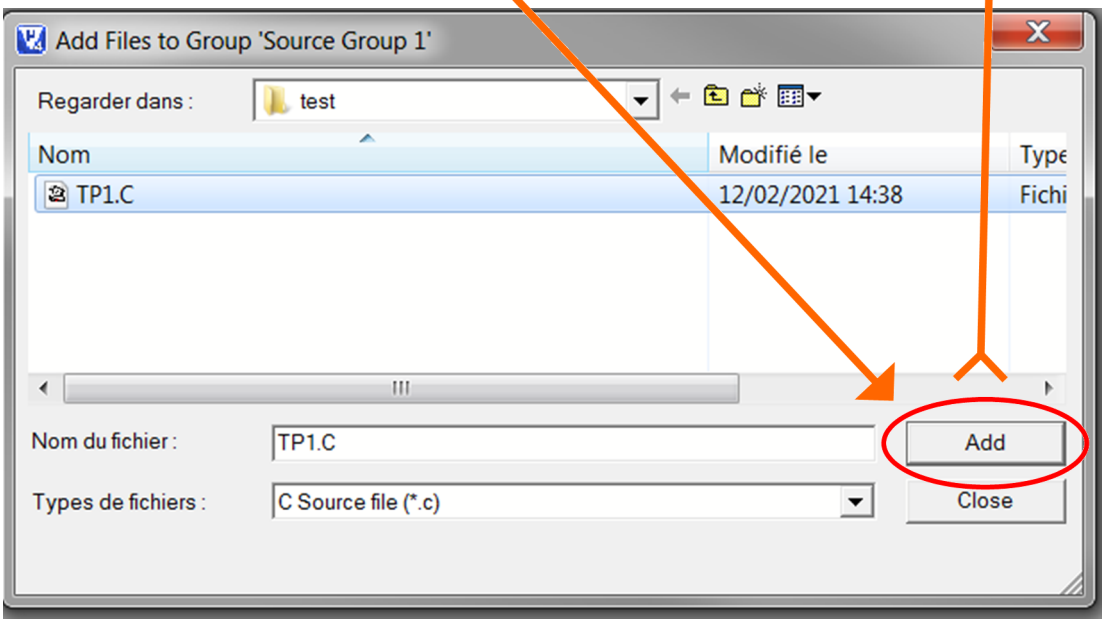
avant

si on compile, on a toujours l'erreur du main manquant



après

si on compile, on n'a plus l'erreur du main manquant



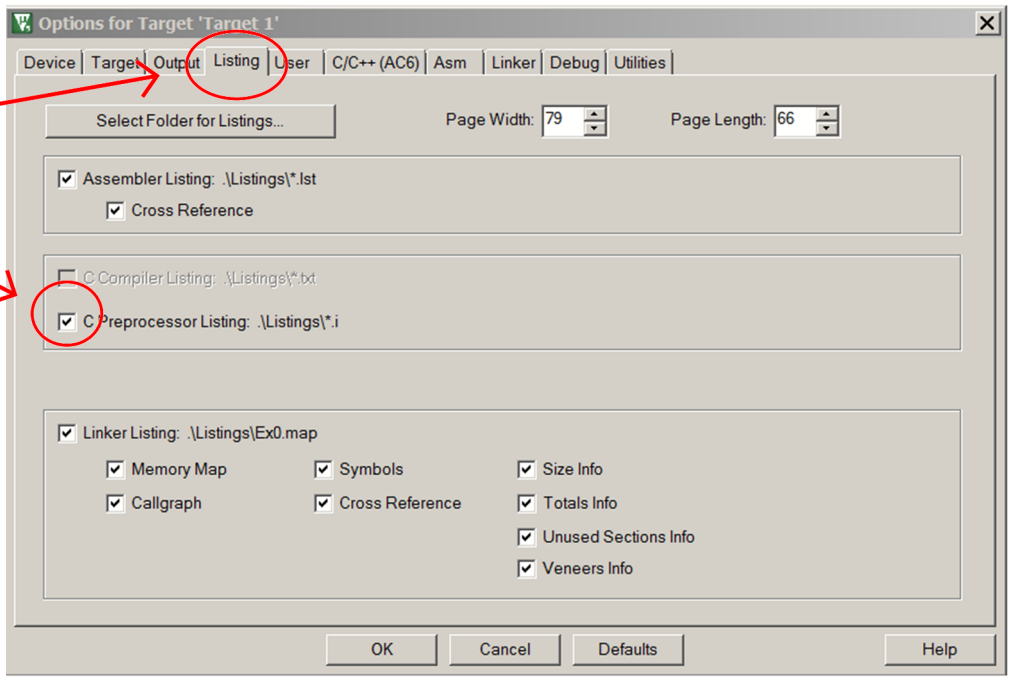
on peut en avoir d'autres si on a mal recopié ... regarder les mots soulignés en rouge dans le code (explications page suivante sur les erreurs)

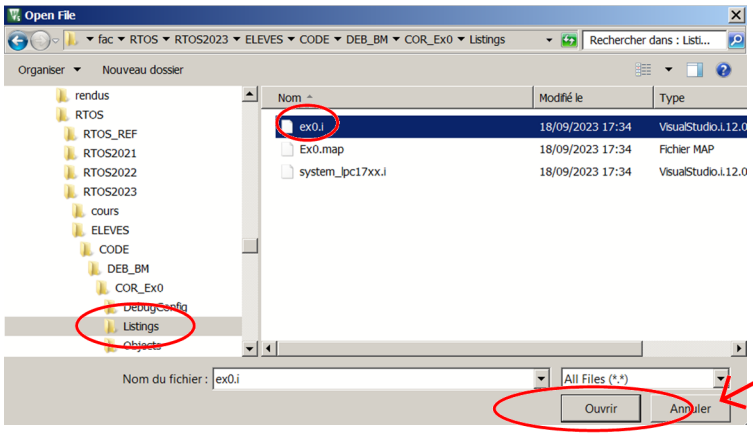
CHAINE DE COMPILATION : PRÉ-COMPILATION

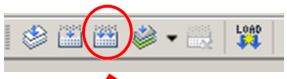
Faire ALT F7 pour faire apparaître le menu OPTIONS FOR TARGET

Dans l'onglet Listing

cochez cette case pour sauvegarder le résultat de la pré-compilation dans un fichier .i







après avoir compilé, ouvrir le fichier .i généré

```
//variables globales
unsigned char  mavar8bits =0;
short mavar16bits = 0;
int  mavar32bits =0;
#define CAS_8 0
#define CAS_16 1
#define CAS_32 2
void init_proc(void)
{
}
void inc_alterne(void)
{//variable statique
 static unsigned char mae=0;
 switch (mae)
  {case CAS_8: mavar8bits++; break;
   case CAS_16: mavar16bits++; break;
   case CAS_32: mavar32bits++; break;
 }
  mae++; if (mae ==3) {mae=0;}
}
int main(void)
{//variable locale
 unsigned char test=0;

init_proc();
 while(1)
  {test++;inc_alterne();
  }
}
```

CODE C

Que deviennent ?

-- Les commentaires

-- Les #define

-- Le code C

A TESTER :

rajouter une macro

#define ADD(a,b) a+b

et le code

test = ADD(test, test);

```
unsigned char  mavar8bits =0;
short mavar16bits = 0;
int  mavar32bits =0;


void init_proc(void)
{
}
void inc_alterne(void)
{
 static unsigned char mae=0;
 switch (mae)
  {case 0: mavar8bits++; break;
   case 1: mavar16bits++; break;
   case 2: mavar32bits++; break;
 }
  mae++; if (mae ==3) {mae=0;}
}
int main(void)
{
 unsigned char test=0;

init_proc();
 while(1)
  {test++;inc_alterne();
  }
}
```

fichier .i

CARTOGRAPHIE MEMOIRE : FICHIER MAP

Ouvrir dans le répertoire listing le fichier .map du projet

rechercher les éléments ci dessous , certains sont trouvables, d'autres pas

Symbol Name	Value	Ov Type	Size Object(Section)
RESET			
Reset_Handler			
HEAP			
STACK			
main			
init_proc			
inc_alterne			
ADD			
mavar8bits			
mavar16bits			
mavar32bits			
mae			
test			

DEFINIR COMMENT SONT GEREES EN MEMOIRE LES DIVERSES ETIQUETTES

PREMIERS PAS avec le LPC1769

Ecrire un code pour initialiser les 8 bits de poids faible du port2 en sortie,
et dans le main faire compter le port :

```
1  unsigned char test=0;
2  void init_proc(void)
3  {LPC_GPIO2->FIODIR = 0xFF;
4  }
5  int main (void)
6  {
7  init_proc();
8   while(1)
9    { test++;
10     LPC_GPIO2->FIOPIN =test;
11   }
12 }
```

L'environnement de travail nous prévient qu'on va avoir des ennuis à la compilation :
il souligne en rouge et met des croix à tous les endroits suspects....

on va compiler pour avoir confirmation

Taper F7

```
Rebuild target 'Target 1'
gpio.c(3): error: use of undeclared identifier 'LPC_GPIO2'
{LPC_GPIO2->FIODIR = 0xFF;
 ^
gpio.c(10): error: use of undeclared identifier 'LPC_GPIO2'
          LPC_GPIO2->FIOPIN =test;
          ^
2 errors generated.
compiling gpio.c...
compiling system_LPC17xx.c...
assembling startup_LPC17xx.s...
".\Objects\gpio.axf" - 2 Error(s), 0 Warning(s).
Target not created.
```

En cliquant sur les erreurs, on est amené directement sur les lignes qui posent problème

LE COMPILATEUR NE CONNAIT PAS LES REGISTRES ??????

normal : il faut inclure un fichier .h qui les décrit ...

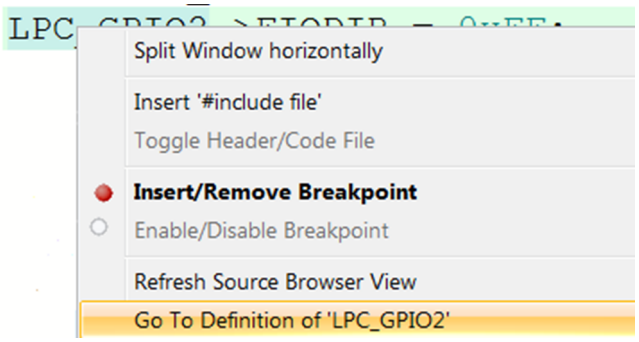
```
1  #include "LPC17xx.h"
2  unsigned char test=0;
3  void init_proc(void)
4  {LPC_GPIO2->FIODIR = 0xFF;
5  }
6  int main (void)
7  {
8  init_proc();
9   while(1)
10   { test++;
11     LPC_GPIO2->FIOPIN =test;
12   }
13 }
```

à l'instant où l'on pose ce " final, tous les signalements d'erreur disparaissent !!!

on va tenter de regarder maintenant le rôle de ce fichier include :

faire un clic droit (clic contextuel) sur FIODIR
choisir Go To Definition Of 'FIODIR'

faire un clic droit (clic contextuel) sur LPC_GPIO2
choisir Go To Definition Of 'LPC_GPIO2'



La compilation a lieu maintenant sans erreurs et sans warning

/** General Purpose Input/Output (GPIO) register structure definition */

en cherchant la déclaration de LPC_GPIO2, on tombe sur une cascade de #define et la définition de la structure à droite

```
#define LPC_GPIO_BASE        (0x2009C000UL)

/* GPIOs    */

#define LPC_GPIO0_BASE       (LPC_GPIO_BASE + 0x00000)
#define LPC_GPIO1_BASE       (LPC_GPIO_BASE + 0x00020)
#define LPC_GPIO2_BASE       (LPC_GPIO_BASE + 0x00040)
#define LPC_GPIO3_BASE       (LPC_GPIO_BASE + 0x00060)
#define LPC_GPIO4_BASE       (LPC_GPIO_BASE + 0x00080)

/*****************************************************************************/
/*                        Peripheral declaration                             */
/*****************************************************************************/
#define LPC_GPIO0 ((LPC_GPIO_TypeDef    *)  LPC_GPIO0_BASE    )
#define LPC_GPIO1 ((LPC_GPIO_TypeDef    *) LPC_GPIO1_BASE    )
#define LPC_GPIO2 ((LPC_GPIO_TypeDef    *) LPC_GPIO2_BASE    )
#define LPC_GPIO3 ((LPC_GPIO_TypeDef    *) LPC_GPIO3_BASE    )
#define LPC_GPIO4 ((LPC_GPIO_TypeDef    *) LPC_GPIO4_BASE    )
```

```
typedef struct
{
  union {
    __IO uint32_t FIODIR;          0x00
    struct {
      __IO uint16_t FIODIRL;       0x00
      __IO uint16_t FIODIRH;       0x02
    };
    struct {
      __IO uint8_t  FIODIR0;       0x00
      __IO uint8_t  FIODIR1;       0x01
      __IO uint8_t  FIODIR2;       0x02
      __IO uint8_t  FIODIR3;       0x03
    };
  };
  uint32_t RESERVED0[3];
  union {
    __IO uint32_t FIOMASK;         0x10
    struct {
      __IO uint16_t FIOMASKL;
      __IO uint16_t FIOMASKH;
    };
    struct {
      __IO uint8_t  FIOMASK0;
      __IO uint8_t  FIOMASK1;
      __IO uint8_t  FIOMASK2;
      __IO uint8_t  FIOMASK3;
    };
  };
  union {
    __IO uint32_t FIOPIN;          0x14
    struct {
      __IO uint16_t FIOPINL;       0x14
      __IO uint16_t FIOPINH;       0x16
    };
    struct {
      __IO uint8_t  FIOPIN0;       0x14
      __IO uint8_t  FIOPIN1;       0x15
      __IO uint8_t  FIOPIN2;       0x16
      __IO uint8_t  FIOPIN3;       0x17
    };
  };
  union {
    __IO uint32_t FIOSET;          0x18
    struct {
      __IO uint16_t FIOSETL;       0x18
      __IO uint16_t FIOSETH;       0x1A
    };
    struct {
      __IO uint8_t  FIOSET0;       0x18
      __IO uint8_t  FIOSET1;       0x19
      __IO uint8_t  FIOSET2;       0x1A
      __IO uint8_t  FIOSET3;       0x1B
    };
  };
  union {
    __O  uint32_t FIOCLR;          0x1C
    struct {
      __O  uint16_t FIOCLRL;       0x1C
      __O  uint16_t FIOCLRH;       0x1E
    };
    struct {
      __O  uint8_t  FIOCLR0;       0x1C
      __O  uint8_t  FIOCLR1;       0x1D
      __O  uint8_t  FIOCLR2;       0x1E
      __O  uint8_t  FIOCLR3;       0x1F
    };
  };
} LPC_GPIO_TypeDef;
```

on voit qu'écrire dans un registre :
LPC_GPIO2->FIOSET = 1<<10;
revient à modifier le contenu d'une case mémoire dont l'adresse est calculée par des sommes successives :
adresse de base (0x2009C000 + 0x00000040)
+ décalage lié au fait d'être le au 7ième entier 32 bits d'une structure LPC_GPIO_TypeDef
donc rajouter 0x18 à cette adresse de base

pour avoir l'adresse finale : 0x2009C058
cdans cette case mémoire qu'on viendra mettre la valeur 1<< 10;

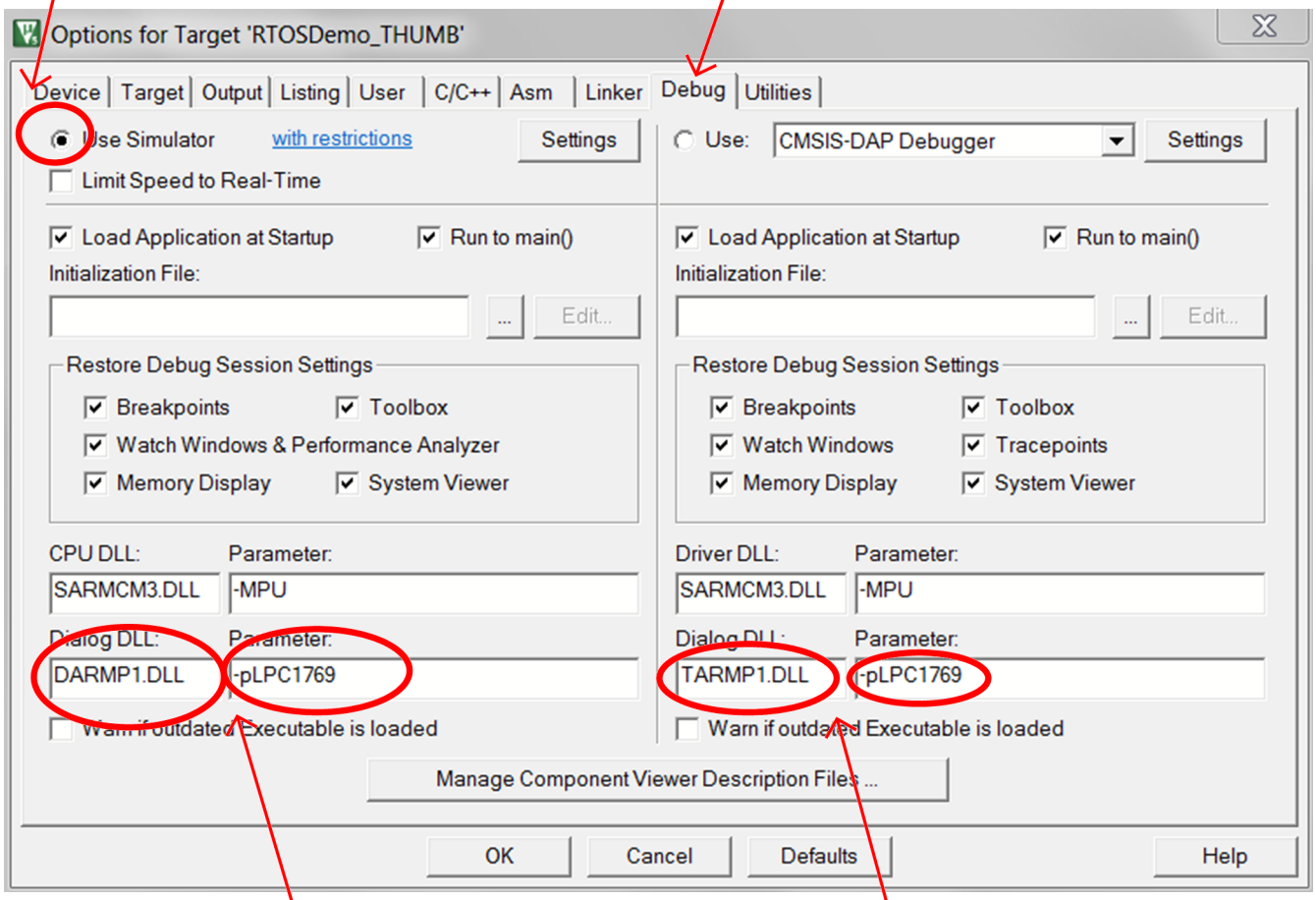
On retrouve cette adresse dans la doc du LPC1769

SANS LE FICHIER "LPC17xx.h" le compilateur n'a aucune information sur ce qu'on souhaite.

Vérifier dans le fichier .l comment l'instruction a été traduite

Pour simuler notre projet, il faut retourner dans options for Target, onglet Debug (ALT F7) debug

cocher Use Simulator



faire très attention à ces champs :
si on voit apparaître DCM.dll et -pCM3 alors la simulation s'arrête dès qu'on tente d'écrire dans un registre propre au LPC1769 car ce simulateur générique des cortexM3 n'a pas les droits d'écriture aux adresses des registres du LPC1769.

faire très attention à ces champs :
si on voit apparaître TCM.dll et -pCM3 alors le debug sur cible s'arrête dès qu'on tente d'écrire dans un registre propre au LPC1769 car ce debugger générique des cortexM3 n'a pas les droits d'écriture aux adresses des registres du LPC1769.

Si les champs sont erronés , corriger à la main

ANNEXE 1

comparatif détaillé des options de compilation :

Options de compilation en version 6 sans avoir basculé au préalable en version 5

```
-xc -std=c99 --target=arm-arm-none-eabi -mcpu=cortex-m3 -c
-fno-rtti -funsigned-char -fshort-enums -fshort-wchar
-D__EVAL -gdwarf-3 -O1 -ffunction-sections -Weverything -Wno-packed -Wno-reserved-id-macro -Wno-unused-macros
-Wno-documentation-unknown-command -Wno-documentation -Wno-license-management -Wno-parentheses-equality
-I./RTE/Device/LPC1769
-I./RTE/_Target_1
-IC:/Users/francis/AppData/Local/Arm/Packs/ARM/CMSIS/5.7.0/CMSIS/Core/Include
-IC:/Users/francis/AppData/Local/Arm/Packs/Keil/LPC1700_DFP/2.6.0/Device/Include
-D__UVISION_VERSION="531" -D_RTE_ -DLPC175x_6x -D_RTE_
-o ./Objects/*.o -MD
```

Options de compilation après avoir basculé en version 5

```
-c --cpu Cortex-M3 -D__EVAL -g -O0 --apcs=interwork --split_sections
-IC:/Keil_v5/ARM/RV31/INC
-IC:/Keil_v5/ARM/CMSIS/Include
-IC:/Users/francis/AppData/Local/Arm/Packs/Keil/LPC1700_DFP/2.6.0/Device/Include
-D__UVISION_VERSION="531"
-o .\Objects\*.o --omf_browse .\Objects\*.crf --depend .\Objects\*.d
```

Options de compilation en version 6 après avoir basculé au préalable en version 5

```
-xc -std=c99 --target=arm-arm-none-eabi -mcpu=cortex-m3 -c
-fno-rtti -funsigned-char -fshort-enums -fshort-wchar
-D__EVAL -gdwarf-3 -O1 -ffunction-sections -Wno-packed -Wno-missing-variable-declarations -Wno-missing-prototypes
-Wno-missing-noreturn -Wno-sign-conversion -Wno-nonportable-include-path -Wno-reserved-id-macro -Wno-unused-macros
-Wno-documentation-unknown-command -Wno-documentation -Wno-license-management -Wno-parentheses-equality
-I./RTE/Device/LPC1769
-I./RTE/_Target_1
-IC:/Users/francis/AppData/Local/Arm/Packs/ARM/CMSIS/5.7.0/CMSIS/Core/Include
-IC:/Users/francis/AppData/Local/Arm/Packs/Keil/LPC1700_DFP/2.6.0/Device/Include
-D__UVISION_VERSION="531" -D_RTE_ -DLPC175x_6x -D_RTE_
-o ./Objects/*.o -MD
```

une simple bascule dans le menu Target vers :

Use default compiler version 5

avec un retour immédiat en :

Use default compiler version 6

fait disparaitre l'option -Weverything qui a déclenché une rafale de mises en garde.... (les 25 warnings)