
Compte-Rendu TP2 :

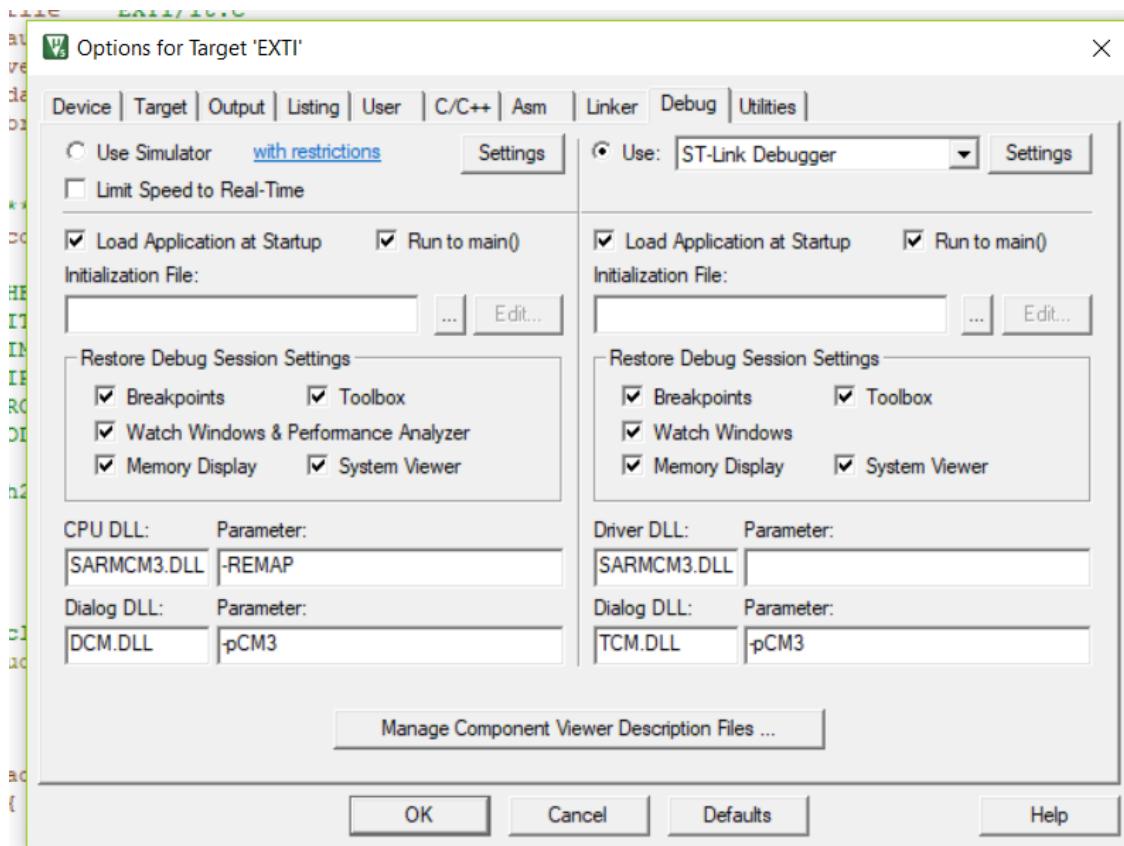
Traitement des interruptions externes sur STM32F1

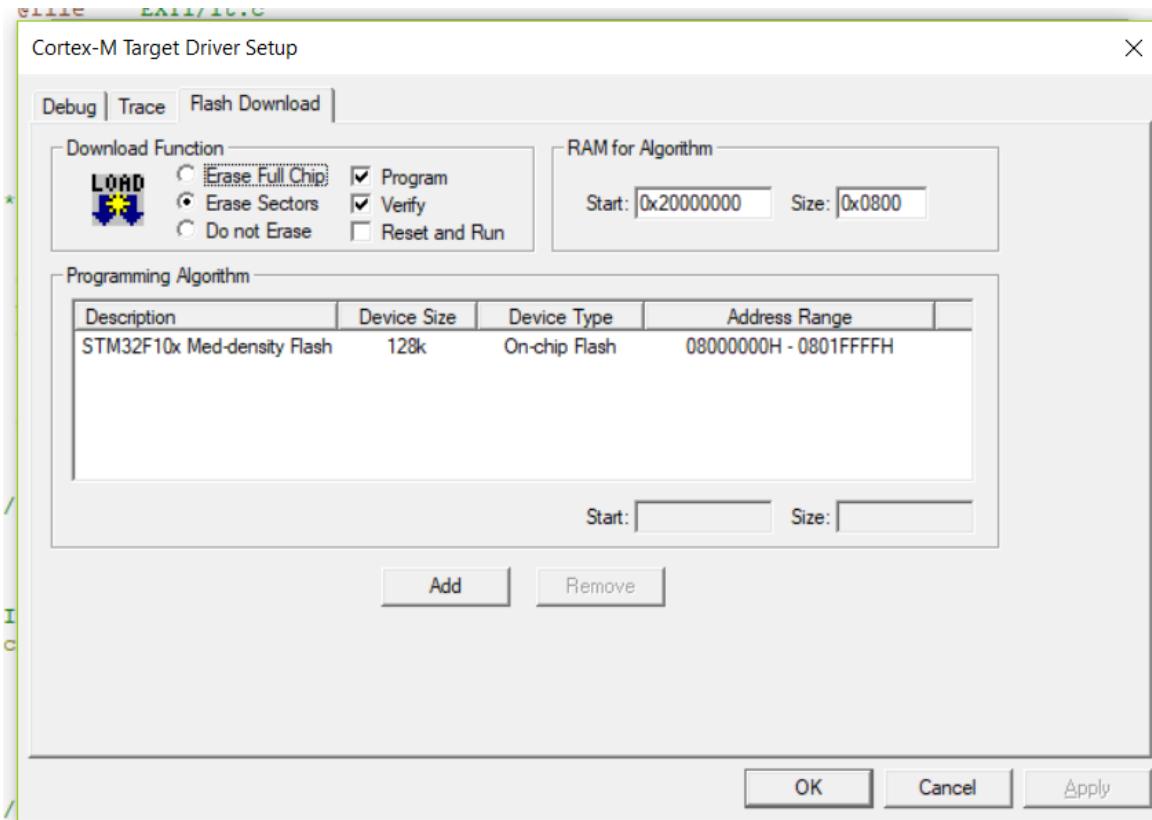
LOUHICHI Najmeddine

ATTIA Malek

Classe : II2D

1. Configuration du projet :





2. Développement du code :

- Les leds sont connectées aux pins 8 et 9 du GPIO C (PC8 et PC9) et le bouton B1 est connecté au pin 0 du GPIO A (PA0).
- Les GPIOs GPIO A et GPIO C sont connectés aux bus APB2.

● GPIO :

- La fonction GPIO_Init() sert à initialiser et écrire les données passées en paramètre dans le GPIO.
- La fonction GPIO_EXTILineConfig() sert à choisir un pin d'un GPIO et utiliser la ligne EXTI associée.

● EXTI :

- On remplit la structure EXTI_InitStructure.
- Les différents champs sont: EXTI_Line: on précise la ligne, EXTI_Mode: on précise le mode et comment l'appui sur le bouton sera perçu, EXTI_Trigger: permet de préciser avec quel front synchroniser le signal de déclenchement, EXTI_LigneCmd: on précise l'état des lignes EXTI sélectionnées.

- Contenu de chaque champs dans la capture.
- La fonction EXTI_Init() est analogue à la fonction GPIO_Init(), elle permet d'initialiser le périphérique EXTI en fonction des valeurs passées en paramètres dans la structure.

● NVIC :

- La structure à remplir est NVIC_InitStructure.
- <<typedef enum IRQn>> représente la définition du numéro d'interruption STM32F10x.
- Il faut remplir le champ NVIC_IRQChannel par EXTI0_IRQHandler et le champ NVIC_IRQChannelCmd par ENABLE.
- La boucle while fait appel à SetBits pour allumer la led attachée au pin 8 du GPIO C

● Main.c

```

46 L
47 int main(void)
48 {
49
50     GPIO_InitTypeDef GPIO_InitStructure;
51     EXTI_InitTypeDef EXTI_InitStructure;
52     NVIC_InitTypeDef NVIC_InitStructure;
53
54     /*!< At this stage the microcontroller clock setting is already configured,
55      this is done through SystemInit() function which is called from startup
56      file (startup_stm32f10x_xx.s) before to branch to application main.
57      To reconfigure the default setting of SystemInit() function, refer to
58      system_stm32f10x.c file
59 */
60
61     /* Create function called "void AllGPIO_Ini(void)" to Configure all unused GPIO port pins in Analog Input mode (floating input
62      trigger OFF), this will reduce the power consumption and increase the device
63      immunity against EMI/EMC *****/
64
65     //AllGPIO_Ini();
66     // attribute clock to all needed peripheral
67     //
68     RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOC|RCC_APB2Periph_GPIOA,ENABLE);
69
70     /* Configure pin led as output mode */
71     GPIO_InitStructure.GPIO_Pin = GPIO_Pin_8|GPIO_Pin_9;           /*!< Specifies the GPIO pins to be configured.
72                                         This parameter can be any value of @ref GPIO_pins_define */
73
74     GPIO_InitStructure.GPIO_Speed= GPIO_Speed_50MHz; /*!< Specifies the speed for the selected pins.
75                                         This parameter can be a value of @ref GPIOSpeed_TypeDef */
76
77     GPIO_InitStructure.GPIO_Mode= GPIO_Mode_Out_PP;
78     GPIO_Init(GPIOC,&GPIO_InitStructure);
79

```

```

79
80     /* Configure PA0 as input floating mode */
81
82     GPIO_InitStructure.GPIO_Pin= GPIO_Pin_0;
83
84     GPIO_InitStructure.GPIO_Mode= GPIO_Mode_IN_FLOATING;
85     GPIO_Init(GPIOA,&GPIO_InitStructure);
86     /* Connect Button EXTI Line to Button GPIO Pin */
87     GPIO_EXTILineConfig(GPIO_PortSourceGPIOA , GPIO_PinSource0);
88
89     /* Configure Button EXTI line */
90     EXTI_InitStructure.EXTI_Line = ENABLE;
91     EXTI_InitStructure.EXTI_Mode = EXTI_Mode_Interrupt;
92     EXTI_InitStructure.EXTI_Trigger = EXTI_Trigger_Rising;
93     EXTI_InitStructure.EXTI_LineCmd = ENABLE;
94     EXTI_Init(&EXTI_InitStructure);
95
96     /* Enable and set Button EXTI Interrupt to the lowest priority */
97     NVIC_InitStructure.NVIC_IRQChannel = EXTI0_IRQn      ;
98     NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0x0F;
99     NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0x0F;
100    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
101    NVIC_Init(&NVIC_InitStructure);
102
103
104
105    while (1)
106    {
107        GPIO_SetBits(GPIOC, GPIO_Pin_8);
108    }
109}
110
111
112 #ifndef USE_FULL_ASSERT

```

- Stm32f10x_it.c
- On utilise la fonction GPIO_ReadInputDataBit() pour voir l'état de la led.
- Le pending bit indique si le code d'une interruption est en train d'attendre d'être exécuté.
- On remet le pending bit à zero pour éviter que le code s'exécute plusieurs fois.

```

142  /*
143   * ***** STM32F10x Peripherals Interrupt Handlers *****
144   */
145  /*
146
147  /**
148   * @brief This function handles External line0 interrupt request.
149   * @param None
150   * @retval None
151   */
152  void EXTI0_IRQHandler(void)
153  {
154
155      if(EXTI_GetITStatus(EXTI_Line0) != RESET)
156      {
157          /* Toggle LED3; we suggest to use GPIO_ReadInputDataBit() function */
158          uint8_t i=GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_9);
159          if(i==0x00){
160              GPIO_SetBits(GPIOC, GPIO_Pin_9);
161          }
162          else{
163              GPIO_ResetBits(GPIOC, GPIO_Pin_9);
164          }
165          /* Clear the User Button EXTI line pending bit */
166          EXTI_ClearITPendingBit(EXTI_Line0);
167      }
168  }
169
170  /*
171   * ***** STM32F10x Peripherals Interrupt Handlers *****
172   * Add here the Interrupt Handler for the used peripheral(s) (PPP), for the
173   * available peripheral interrupt handler's name please refer to the startup
174   * file (startup_stm32f10x_xx.s).
175  */

```

