PSI3441 - Arquitetura de Sistemas Embarcados

Relatório Exercício Prático 06

Nome: Kevin Kirsten Lucas nºUSP: 10853306

1) Utilizar um timer periódico para, por interrupção, disparar a conversão AD por hardware. Usar a interrupção de fim de conversão para acender os LEDs como feito no exercício 7.

De maneira semelhante ao exercício 5 anterior, precisamos configurar o ADC, o Timer e os LEDs. Os códigos gerados estão abaixo:

```c
/* ###################################################################
**     Filename    : main.c
**     Project     : ex_5_3
**     Processor   : MKL25Z128VLK4
**     Version     : Driver 01.01
**     Compiler    : GNU C Compiler
**     Date/Time   : 2022-07-07, 01:35, # CodeGen: 0
**     Abstract    :
**         Main module.
**         This module contains user's application code.
**     Settings    :
**     Contents    :
**         No public methods
**
** ###################################################################*/
/*!
** @file main.c
** @version 01.01
** @brief
**         Main module.
**         This module contains user's application code.
*/
/*!
**  @addtogroup main_module main module documentation
**  @{
*/
/* MODULE main */


/* Including needed modules to compile this module/procedure */
#include "Cpu.h"
#include "Events.h"
#include "TI1.h"
#include "TU1.h"
#include "Bit1_Green_LED.h"
#include "BitIoLdd1.h"
#include "Bit2_Blue_LED.h"
#include "BitIoLdd2.h"
#include "AD1.h"
#include "AdcLdd1.h"
/* Including shared modules, which are used for whole project */
#include "PE_Types.h"
#include "PE_Error.h"
#include "PE_Const.h"
#include "IO_Map.h"

uint16_t adc_value;

/* User includes (#include below this line is not maintained by Processor Expert) */

/*lint -save  -e970 Disable MISRA rule (6.3) checking. */
int main(void)
/*lint -restore Enable MISRA rule (6.3) checking. */
{
  /* Write your local variable definition here */

  /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!! ***/
  PE_low_level_init();
  /*** End of Processor Expert internal initialization.                    ***/

  /* Write your code here */
  Bit1_Green_LED_SetVal();

  /*** Don't write any code pass this line, or it will be deleted during code generation. ***/
  /*** RTOS startup code. Macro PEX_RTOS_START is defined by the RTOS component. DON'T MODIFY THIS CODE!!! ***/
  #ifdef PEX_RTOS_START
    PEX_RTOS_START();                  /* Startup of the selected RTOS. Macro is defined by the RTOS component. */
  #endif
  /*** End of RTOS startup code.  ***/
  /*** Processor Expert end of main routine. DON'T MODIFY THIS CODE!!! ***/
  for(;;){}
  /*** Processor Expert end of main routine. DON'T WRITE CODE BELOW!!! ***/
} /*** End of main routine. DO NOT MODIFY THIS TEXT!!! ***/

/* END main */
/*!
** @}
*/
/*
** ###################################################################
**
**     This file was created by Processor Expert 10.3 [05.09]
**     for the Freescale Kinetis series of microcontrollers.
**
** ###################################################################
*/
```

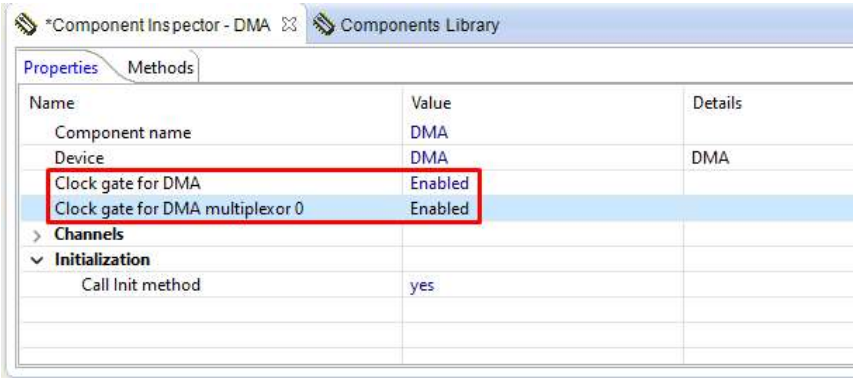
Código do arquivo main.c

```c
/* ###################################################################
**     Filename    : main.c
**     Project     : ex_5_3
**     Processor   : MKL25Z128VLK4
**     Version     : Driver 01.01
**     Compiler    : GNU C Compiler
**     Date/Time   : 2022-07-07, 01:35, # CodeGen: 0
**     Abstract    :
**         Main module.
**         This module contains user's application code.
**     Settings    :
**     Contents    :
**         No public methods
**
** ###################################################################*/
/*!
** @file main.c
** @version 01.01
** @brief
**         Main module.
**         This module contains user's application code.
*/
/*!
**  @addtogroup main_module main module documentation
**  @{
*/
/* MODULE main */


/* Including needed modules to compile this module/procedure */
#include "Cpu.h"
#include "Events.h"
#include "TI1.h"
#include "TU1.h"
#include "Bit1_Green_LED.h"
#include "BitIoLdd1.h"
#include "Bit2_Blue_LED.h"
#include "BitIoLdd2.h"
#include "AD1.h"
#include "AdcLdd1.h"
/* Including shared modules, which are used for whole project */
#include "PE_Types.h"
#include "PE_Error.h"
#include "PE_Const.h"
#include "IO_Map.h"

uint16_t adc_value;

/* User includes (#include below this line is not maintained by Processor Expert) */

/*lint -save  -e970 Disable MISRA rule (6.3) checking. */
int main(void)
/*lint -restore Enable MISRA rule (6.3) checking. */
{
  /* Write your local variable definition here */

  /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!! ***/
  PE_low_level_init();
  /*** End of Processor Expert internal initialization.                    ***/

  /* Write your code here */
  Bit1_Green_LED_SetVal();

  /*** Don't write any code pass this line, or it will be deleted during code generation. ***/
  /*** RTOS startup code. Macro PEX_RTOS_START is defined by the RTOS component. DON'T MODIFY THIS CODE!!! ***/
  #ifdef PEX_RTOS_START
    PEX_RTOS_START();                  /* Startup of the selected RTOS. Macro is defined by the RTOS component. */
  #endif
  /*** End of RTOS startup code.  ***/
  /*** Processor Expert end of main routine. DON'T MODIFY THIS CODE!!! ***/
  for(;;){}
  /*** Processor Expert end of main routine. DON'T WRITE CODE BELOW!!! ***/
} /*** End of main routine. DO NOT MODIFY THIS TEXT!!! ***/

/* END main */
/*!
** @}
*/
/*
** ###################################################################
**
**     This file was created by Processor Expert 10.3 [05.09]
**     for the Freescale Kinetis series of microcontrollers.
**
** ###################################################################
*/

```

Código do arquivo events.c

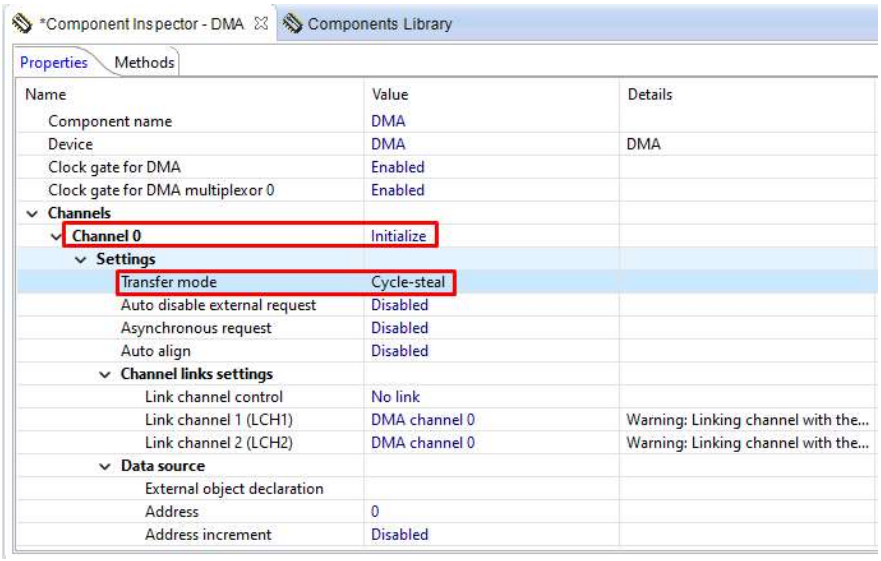
2) utilize DMA para gravar os dados na memória.

Para o DMA, vamos criar um componente `Init_DMA` e realizar as configurações conforme a imagem abaixo:



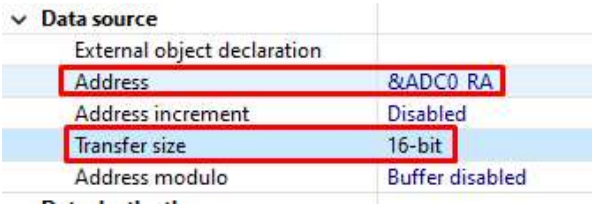
Configuração do Init_DMA

Em seguida, inicializamos o canal 0:



Inicialização do canal 0

Realizamos as configurações para Data source

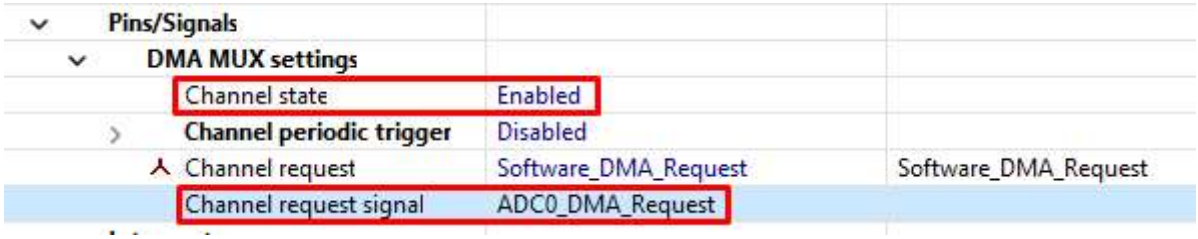


Configuração do data source

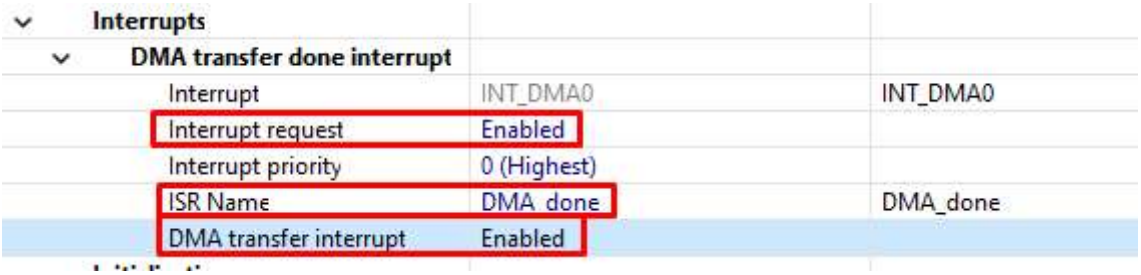
Depois, as configurações para Data destination e Byte count:

Configuração do data destination

Fazemos também a configuração do DMA:

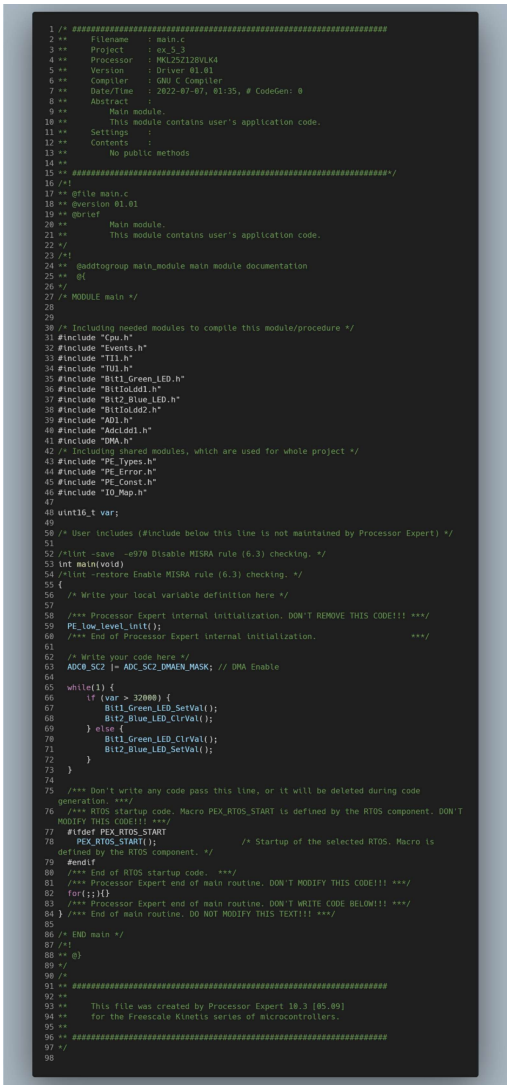


Configuração DMA



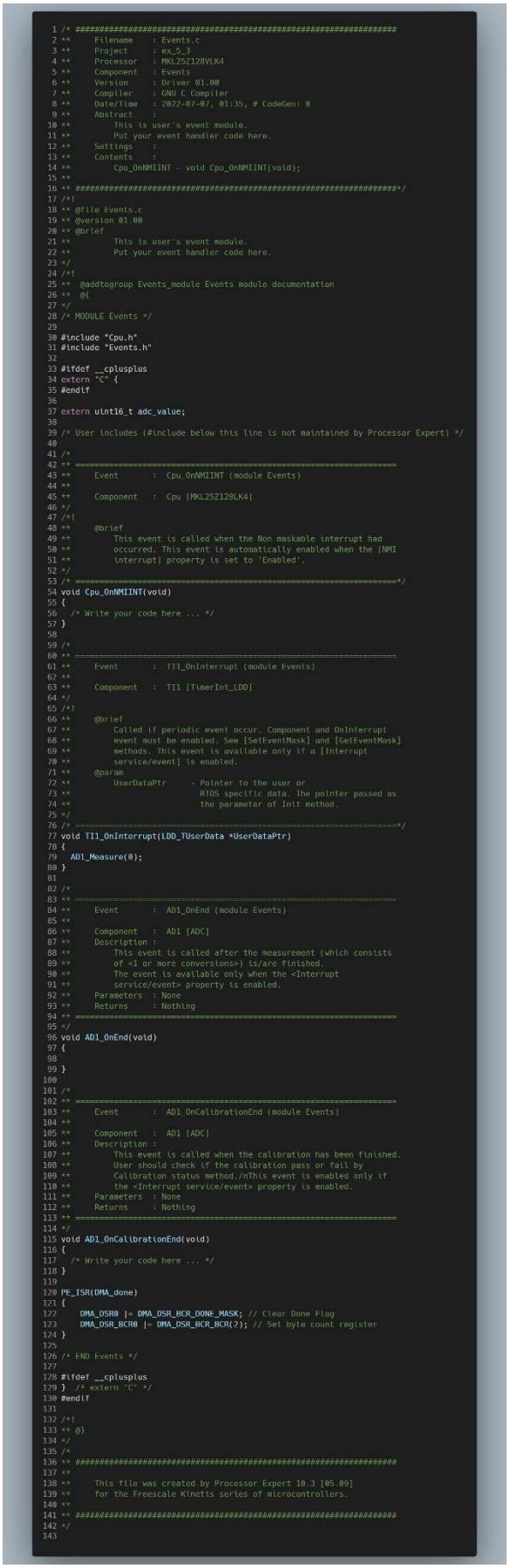
Configuração DMA

Por fim, basta realizar algumas alterações no `main.c`, `events.c` e `DMA.c` conforme mostrado a seguir:



```
/* ###################################################################
**     Filename    : main.c
**     Project     : ex_5_3
**     Processor   : MKL25Z128VLK4
**     Version     : Driver 01.01
**     Compiler    : GNU C Compiler
**     Date/Time   : 2022-07-07, 01:35, # CodeGen: 0
**     Abstract    :
**         Main module.
**         This module contains user's application code.
**     Settings    :
**     Contents    :
**         No public methods
**
** ###################################################################*/
/*!
** @file main.c
** @version 01.01
** @brief
**         Main module.
**         This module contains user's application code.
*/
/*!
**  @addtogroup main_module main module documentation
**  @{
*/
/* MODULE main */


/* Including needed modules to compile this module/procedure */
#include "Cpu.h"
#include "Events.h"
#include "TI1.h"
#include "TU1.h"
#include "Bit1_Green_LED.h"
#include "BitIoLdd1.h"
#include "Bit2_Blue_LED.h"
#include "BitIoLdd2.h"
#include "AD1.h"
#include "AdcLdd1.h"
#include "DMA.h"
/* Including shared modules, which are used for whole project */
#include "PE_Types.h"
#include "PE_Error.h"
#include "PE_Const.h"
#include "IO_Map.h"

uint16_t var;

/* User includes (#include below this line is not maintained by Processor Expert) */

/*lint -save  -e970 Disable MISRA rule (6.3) checking. */
int main(void)
/*lint -restore Enable MISRA rule (6.3) checking. */
{
  /* Write your local variable definition here */

  /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!! ***/
  PE_low_level_init();
  /*** End of Processor Expert internal initialization.                    ***/

  /* Write your code here */
  ADC0_SC2 |= ADC_SC2_DMAEN_MASK; // DMA Enable

  while(1) {
      if (var > 32000) {
          Bit1_Green_LED_SetVal();
          Bit2_Blue_LED_ClrVal();
      } else {
          Bit1_Green_LED_ClrVal();
          Bit2_Blue_LED_SetVal();
      }
  }

  /*** Don't write any code pass this line, or it will be deleted during code
  generation. ***/
  /*** RTOS startup code. Macro PEX_RTOS_START is defined by the RTOS component. DON'T
  MODIFY THIS CODE!!! ***/
  #ifdef PEX_RTOS_START
    PEX_RTOS_START();                  /* Startup of the selected RTOS. Macro is
  defined by the RTOS component. */
  #endif
  /*** End of RTOS startup code.  ***/
  /*** Processor Expert end of main routine. DON'T MODIFY THIS CODE!!! ***/
  for(;;){}
  /*** Processor Expert end of main routine. DON'T WRITE CODE BELOW!!! ***/
} /*** End of main routine. DO NOT MODIFY THIS TEXT!!! ***/

/* END main */
/*!
** @}
*/
/*
** ###################################################################
**
**     This file was created by Processor Expert 10.3 [05.09]
**     for the Freescale Kinetis series of microcontrollers.
**
** ###################################################################
*/

```

Código do arquivo main.c



```c
/* ###################################################################
**     Filename    : Events.c
**     Project     : ex_5_3
**     Processor   : MKL25Z128VLK4
**     Component   : Events
**     Version     : Driver 01.00
**     Compiler    : GNU C Compiler
**     Date/Time   : 2022-07-07, 01:35, # CodeGen: 0
**     Abstract    :
**         This is user's event module.
**         Put your event handler code here.
**     Settings    :
**     Contents    :
**         Cpu_OnNMIINT - void Cpu_OnNMIINT(void);
**
** ###################################################################*/
/*!
** @file Events.c
** @version 01.00
** @brief
**         This is user's event module.
**         Put your event handler code here.
*/
/*!
**  @addtogroup Events_module Events module documentation
**  @{
*/
/* MODULE Events */

#include "Cpu.h"
#include "Events.h"

#ifdef __cplusplus
extern "C" {
#endif

extern uint16_t adc_value;

/* User includes (#include below this line is not maintained by Processor Expert) */

/*
** ===================================================================
**     Event       :  Cpu_OnNMIINT (module Events)
**
**     Component   :  Cpu [MKL25Z128LK4]
*/
/*!
**     @brief
**         This event is called when the Non maskable interrupt had
**         occurred. This event is automatically enabled when the [NMI
**         interrupt] property is set to 'Enabled'.
*/
/* ===================================================================*/
void Cpu_OnNMIINT(void)
{
  /* Write your code here ... */
}

/*
** ===================================================================
**     Event       :  TI1_OnInterrupt (module Events)
**
**     Component   :  TI1 [TimerInt_LDD]
*/
/*!
**     @brief
**         Called if periodic event occur. Component and OnInterrupt
**         event must be enabled. See [SetEventMask] and [GetEventMask]
**         methods. This event is available only if a [Interrupt
**         service/event] is enabled.
**     @param
**         UserDataPtr     - Pointer to the user or
**                           RTOS specific data. The pointer passed as
**                           the parameter of Init method.
*/
/* ===================================================================*/
void TI1_OnInterrupt(LDD_TUserData *UserDataPtr)
{
  AD1_Measure(0);
}

/*
** ===================================================================
**     Event       :  AD1_OnEnd (module Events)
**
**     Component   :  AD1 [ADC]
**     Description :
**         This event is called after the measurement (which consists
**         of <1 or more conversions>) is/are finished.
**         The event is available only when the <Interrupt
**         service/event> property is enabled.
**     Parameters  : None
**     Returns     : Nothing
** ===================================================================
*/
void AD1_OnEnd(void)
{

}

/*
** ===================================================================
**     Event       :  AD1_OnCalibrationEnd (module Events)
**
**     Component   :  AD1 [ADC]
**     Description :
**         This event is called when the calibration has been finished.
**         User should check if the calibration pass or fail by
**         Calibration status method./nThis event is enabled only if
**         the <Interrupt service/event> property is enabled.
**     Parameters  : None
**     Returns     : Nothing
** ===================================================================
*/
void AD1_OnCalibrationEnd(void)
{
  /* Write your code here ... */
}

PE_ISR(DMA_done)
{
    DMA_DSR0 |= DMA_DSR_BCR_DONE_MASK; // Clear Done Flag
    DMA_DSR_BCR0 |= DMA_DSR_BCR_BCR(2); // Set byte count register
}

/* END Events */

#ifdef __cplusplus
}  /* extern "C" */
#endif

/*!
** @}
*/
/*
** ###################################################################
**
**     This file was created by Processor Expert 10.3 [05.09]
**     for the Freescale Kinetis series of microcontrollers.
**
** ###################################################################
*/
```

Código do arquivo events.c

Código do arquivo DMA.c

Link com o código dos arquivos do projeto: https://github.com/kevinkirsten/psi3441-arquitetura-de-sistemas-embarcados/tree/main/exercicio_pratico_06