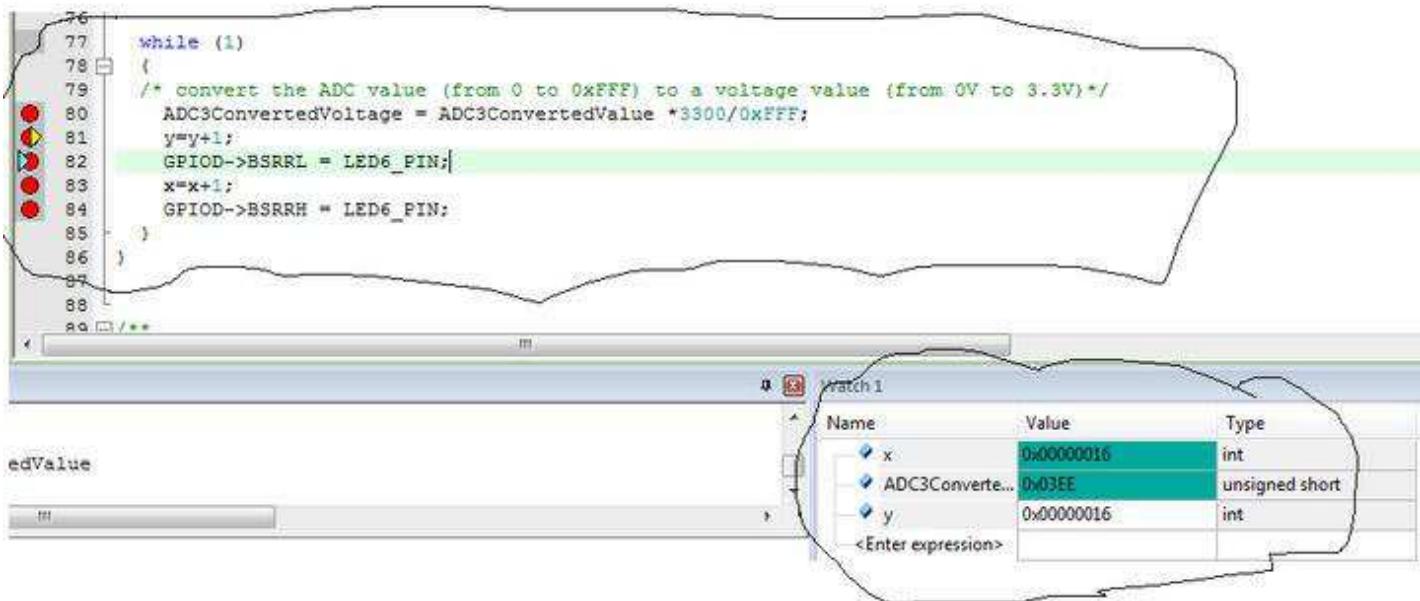


Hi,

I bought a development board STM32F4 discovery.

I wanted to try the debug and I modified the example ADC3_DMA in STM32F4-Discovery_FW_V1.1.0 in this mode:



but the program does not stop never on the breakpoint at lines 80, 82 and 84 (`ADC3ConvertedVoltage = ADC3ConvertedValue *3300/0xFFFF; GPIOD->BSRRL = LED6_PIN; GPIOD->BSRRH = LED6_PIN;`) but only at lines 81 and 83 (`x=x+1; y=y+1;`)
has someone already had a similar problem? and can give me suggestions?

thanks

if shoud to be help I have attached the main

```

/**
 ****
 * @file      ADC3_DMA/main.c
 * @author    MCD Application Team
 * @version   V1.0.0
 * @date      19-September-2011
 * @brief     Main program body
 ****
 * @attention
 *
 * THE PRESENT FIRMWARE WHICH IS FOR GUIDANCE ONLY AIMS AT PROVIDING CUSTOMERS
 * WITH CODING INFORMATION REGARDING THEIR PRODUCTS IN ORDER FOR THEM TO SAVE
 * TIME. AS A RESULT, STMICROELECTRONICS SHALL NOT BE HELD LIABLE FOR ANY
 * DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WITH RESPECT TO ANY CLAIMS ARISING
 * FROM THE CONTENT OF SUCH FIRMWARE AND/OR THE USE MADE BY CUSTOMERS OF THE
 * CODING INFORMATION CONTAINED HEREIN IN CONNECTION WITH THEIR PRODUCTS.
 *
 * <h2><center>© COPYRIGHT 2011 STMicroelectronics</center></h2>
 ****
 */

/* Includes ----- */
#include "stm32f4_discovery.h"
#include <stdio.h>

/** @addtogroup STM32F4_Discovery_Peripheral_Examples
 * @{
 */
    
```

```
/** @addtogroup ADC_ADC3_DMA
 * @{
 */

/* Private typedef -----*/
/* Private define -----*/
#define ADC3_DR_ADDRESS ((uint32_t)0x4001224C)

/* Private macro -----*/
/* Private variables -----*/
/* You can monitor the converted value by adding the variable "ADC3ConvertedValue"
   to the debugger watch window */
__IO uint16_t ADC3ConvertedValue = 0;
__IO uint32_t ADC3ConvertedVoltage = 0;
int x;
int y;

/* Private function prototypes -----*/
/* Private functions -----*/
void ADC3_CH12_DMA_Config(void);
void InizializzaPorta(void);

/***
 * @brief Main program
 * @param None
 * @retval None
 */
int main(void)
{
    /*!< At this stage the microcontroller clock setting is already configured,
        this is done through SystemInit() function which is called from startup
        file (startup_stm32f4xx.s) before to branch to application main.
        To reconfigure the default setting of SystemInit() function, refer to
        system_stm32f4xx.c file
    */
    /* ADC3 configuration *****/
    /* - Enable peripheral clocks */
    /* - DMA2_Stream0 channel2 configuration */
    /* - Configure ADC Channel12 pin as analog input */
    /* - Configure ADC3 Channel12 */
    ADC3_CH12_DMA_Config();

    /* Start ADC3 Software Conversion */
    ADC_SoftwareStartConv(ADC3);
    InizializzaPorta();

    while (1)
    {
        /* convert the ADC value (from 0 to 0xFFFF) to a voltage value (from 0V to 3.3V)*/
        ADC3ConvertedVoltage = ADC3ConvertedValue *3300/0xFFFF;
        y=y+1;
        GPIOD->BSRRL = LED6_PIN;
        x=x+1;
        GPIOD->BSRRH = LED6_PIN;
    }
}

/***
 * @brief ADC3 channel12 with DMA configuration
 * @param None
 * @retval None
 */
void InizializzaPorta(void)
{
    GPIO_InitTypeDef NomePorta;
```

```

/* Abilito il segnale di clock della porta */
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOD, ENABLE);
/* Enable SYSCFG clock */
RCC_APB2PeriphClockCmd(RCC_APB2Periph_SYSCFG, ENABLE);

/* Configura il pin della porta D numero 15 PD15 al quale è collegato il led */
NomePorta.GPIO_Mode = 0x01; // scrivo 0x01 nel registro MODER il che vuol
dire porre il pin in output ---> NomePorta.GPIO_Mode = GPIO_Mode_OUT; vedi manuale STM32F4xxx.PDF
NomePorta.GPIO_OType = 0x00; // scrivo 0x00 nel registro OTYPER il che
vuol dire porre l'output in push pull (se scrivevamo 0x01 lo poneva in open drain) ---
>NomePorta.GPIO_OType =GPIO_OType_PP; vedi manuale STM32F4xxx.PDF
NomePorta.GPIO_Pin = ((uint16_t)0x8000); //GPIO_Pin_12; //non sostituisco con
esadecimale per semplicità
NomePorta.GPIO_PuPd = 0x01; //Imposto il pin in pull up scrivendo nel
registro NomePorta.GPIO_PuPd =GPIO_PuPd_UP
NomePorta.GPIO_Speed = 0x02; //Imposto la velocità del pin ---->
NomePorta.GPIO_Speed = GPIO_Speed_50MHz
GPIO_Init(GPIOD, &NomePorta); //Funzione che va a mettere i valori dei campi
della struttura NomePorta che abbiamo definito nei registri elencati nei commenti precedenti
}

void ADC3_CH12_DMA_Config(void)
{
    ADC_InitTypeDef      ADC_InitStructure;
    ADC_CommonInitTypeDef ADC_CommonInitStructure;
    DMA_InitTypeDef      DMA_InitStructure;
    GPIO_InitTypeDef      GPIO_InitStructure;

    /* Enable ADC3, DMA2 and GPIO clocks *****/
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_DMA2 | RCC_AHB1Periph_GPIOC, ENABLE);
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC3, ENABLE);

    /* DMA2 Stream0 channel0 configuration *****/
    DMA_InitStructure.DMA_Channel = DMA_Channel_2;
    DMA_InitStructure.DMA_PeripheralBaseAddr = (uint32_t)ADC3_DR_ADDRESS;
    DMA_InitStructure.DMA_Memory0BaseAddr = (uint32_t)&ADC3ConvertedValue;
    DMA_InitStructure.DMA_DIR = DMA_DIR_PeripheralToMemory;
    DMA_InitStructure.DMA_BufferSize = 1;
    DMA_InitStructure.DMA_PeripheralInc = DMA_PeripheralInc_Disable;
    DMA_InitStructure.DMA_MemoryInc = DMA_MemoryInc_Disable;
    DMA_InitStructure.DMA_PeripheralDataSize = DMA_PeripheralDataSize_HalfWord;
    DMA_InitStructure.DMA_MemoryDataSize = DMA_MemoryDataSize_HalfWord;
    DMA_InitStructure.DMA_Mode = DMA_Mode_Circular;
    DMA_InitStructure.DMA_Priority = DMA_Priority_High;
    DMA_InitStructure.DMA_FIFOMode = DMA_FIFOMode_Disable;
    DMA_InitStructure.DMA_FIFOThreshold = DMA_FIFOThreshold_HalfFull;
    DMA_InitStructure.DMA_MemoryBurst = DMA_MemoryBurst_Single;
    DMA_InitStructure.DMA_PeripheralBurst = DMA_PeripheralBurst_Single;
    DMA_Init(DMA2_Stream0, &DMA_InitStructure);
    DMA_Cmd(DMA2_Stream0, ENABLE);

    /* Configure ADC3 Channel12 pin as analog input *****/
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_2;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AN;
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL ;
    GPIO_Init(GPIOC, &GPIO_InitStructure);

    /* ADC Common Init *****/
    ADC_CommonInitStructure.ADC_Mode = ADC_Mode_Independent;
    ADC_CommonInitStructure.ADC_Prescaler = ADC_Prescaler_Div2;
    ADC_CommonInitStructure.ADC_DMAAccessMode = ADC_DMAAccessMode_Disabled;
    ADC_CommonInitStructure.ADC_TwoSamplingDelay = ADC_TwoSamplingDelay_5Cycles;
    ADC_CommonInit(&ADC_CommonInitStructure);

    /* ADC3 Init *****/
    ADC_InitStructure.ADC_Resolution = ADC_Resolution_12b;
    ADC_InitStructure.ADC_ScanConvMode = DISABLE;
    ADC_InitStructure.ADC_ContinuousConvMode = ENABLE;
    ADC_InitStructure.ADC_ExternalTrigConvEdge = ADC_ExternalTrigConvEdge_None;
}

```

```
ADC_InitStructure.ADC_DataAlign = ADC_DataAlign_Right;
ADC_InitStructure.ADC_NbrOfConversion = 1;
ADC_Init(ADC3, &ADC_InitStructure);

/* ADC3 regular channel12 configuration *****/
ADC-RegularChannelConfig(ADC3, ADC_Channel_12, 1, ADC_SampleTime_3Cycles);

/* Enable DMA request after last transfer (Single-ADC mode) */
ADC_DMARequestAfterLastTransferCmd(ADC3, ENABLE);

/* Enable ADC3 DMA */
ADC_DMACmd(ADC3, ENABLE);

/* Enable ADC3 */
ADC_Cmd(ADC3, ENABLE);
}

#ifndef USE_FULL_ASSERT

/***
 * @brief Reports the name of the source file and the source line number
 * where the assert_param error has occurred.
 * @param file: pointer to the source file name
 * @param line: assert_param error line source number
 * @retval None
 */
void assert_failed(uint8_t* file, uint32_t line)
{
    /* User can add his own implementation to report the file name and line number,
       ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */

    /* Infinite loop */
    while (1)
    {
    }
}
#endif

/***
 * @}
 */

/***
 * @}
 */

***** (C) COPYRIGHT 2011 STMicroelectronics *****END OF FILE****/
```