Hi,

I am trying to let my STM32F429 Disco Board sample in triple mode interleaved. I mainly copied the code from ST, with small changes, and it also works. But the word-array I want the DMA to transfer the data to, has some bugs.
every element of the word-array aADCTripleConvertedValue[] keeps two ADC results. The lower half word gets the first result, whereas the upper half word gets the second result:
[0]=[ADC2, ADC1]
[1]=[ADC1, ADC3]
[2]=[ADC3, ADC2]

this works fine for the first 6 results. But then I get this:

[3]=[2]=[ADC3, ADC2]
[4]=[2]=[ADC3, ADC2]
[5]=[2]=[ADC3, ADC2]

and after that, I get the new results:

[6]=[ADC2, ADC1]
[7]=[ADC1, ADC3]
[8]=[ADC3, ADC2]

and again:

[9]=[10]=[11]=[8]

if you look at the image, you see that there is no data missing, but it looks as if the DMA copies old values, until it gets the new one. If I delete the copied Data, the 300kHz sine looks fine. Does anybody know whats wrong here?

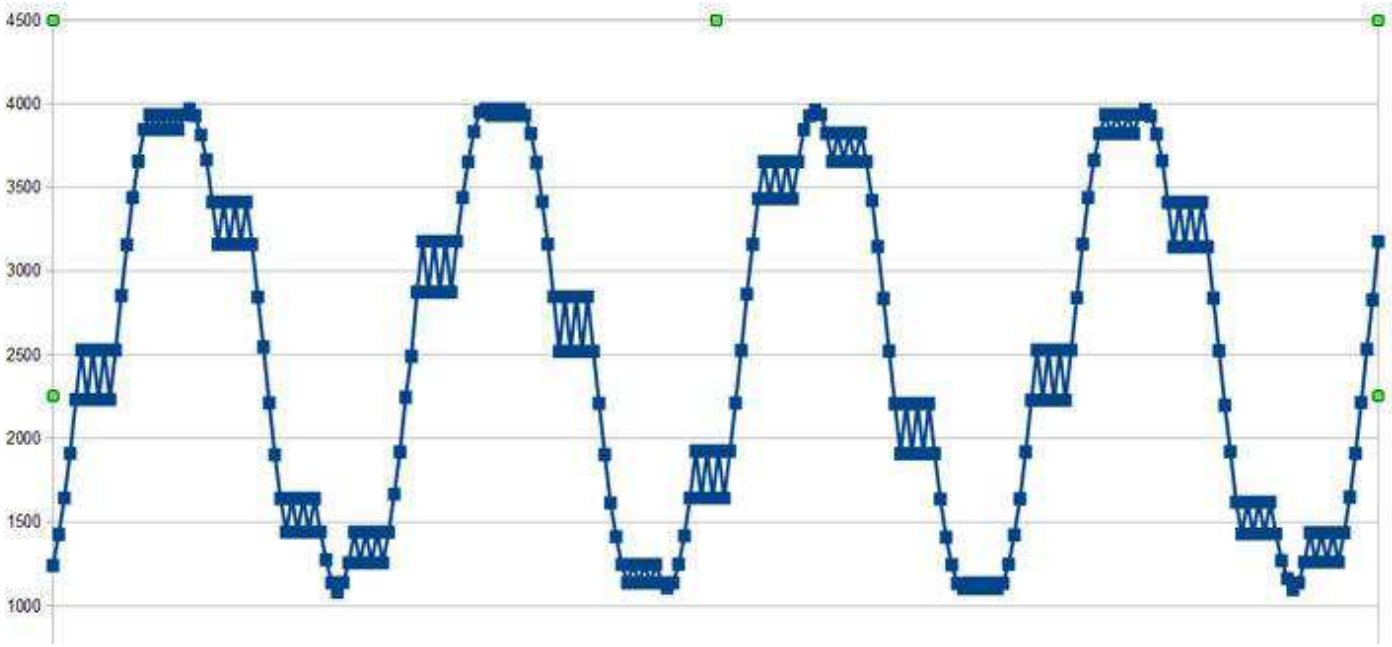
code:

```
/**
  ******************************************************************************
  * @file    ADC/ADC_TripleModeInterleaved/main.c
  * @author  MCD Application Team
  * @version V1.1.0
  * @date    18-January-2013
  * @brief   Main program body
  ******************************************************************************
  * @attention
  *
  * <h2><center>© COPYRIGHT 2013 STMicroelectronics</center></h2>
  *
  * Licensed under MCD-ST Liberty SW License Agreement V2, (the "License");
  * You may not use this file except in compliance with the License.
  * You may obtain a copy of the License at:
  *
  *        http://www.st.com/software_license_agreement_liberty_v2
  *
  * Unless required by applicable law or agreed to in writing, software
  * distributed under the License is distributed on an "AS IS" BASIS,
  * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  * See the License for the specific language governing permissions and
  * limitations under the License.
  *
  ******************************************************************************
  */
/* Includes ------------------------------------------------------------------*/
#include "triple interleaved.h"

#define TripleADCMessungen 120
```

```
/** @addtogroup STM32F4xx_StdPeriph_Examples
  * @{
  */

/** @addtogroup ADC_TripleModeInterleaved
  * @{
  */

/* Private typedef -----------------------------------------------------------*/
/* Private define ------------------------------------------------------------*/
/* Private macro -------------------------------------------------------------*/
/* Private variables ---------------------------------------------------------*/
extern volatile uint32_t aADCTripleConvertedValue[TripleADCMessungen];

/* Private function prototypes -----------------------------------------------*/
static void ADC_Config(void);

/* Private functions ---------------------------------------------------------*/

/**
  * @brief   Main program
  * @param  None
  * @retval None
  */

void tripleADCreset(void){


    DMA_ClearFlag(DMA_STREAMx, DMA_FLAG_TCIF0);
    ADC_DeInit();
    ADC->CCR &= ~((uint32_t)(0b00000));
}

void tripleADCstart(void)
{
  /*!< At this stage the microcontroller clock setting is already configured,
       this is done through SystemInit() function which is called from startup
       files (startup_stm32f40xx.s/startup_stm32f427x.s) before to branch to
       application main.
       To reconfigure the default setting of SystemInit() function, refer to
       system_stm32f4xx.c file
     */

  /* ADC configuration */
  ADC_Config();

  /* Start ADC1 Software Conversion */
  ADC_SoftwareStartConv(ADC1);


}

/**
  * @brief  ADC configuration
  * @note   This function Configure the ADC peripheral
            1) Enable peripheral clocks
            2) Configure ADC Channel 12 pin as analog input
            3) DMA2_Stream0 channel2 configuration
            4) Configure ADC1 Channel 12
            5) Configure ADC2 Channel 12
            6) Configure ADC3 Channel 12
  * @param  None
  * @retval None
  */
static void ADC_Config(void)
{
  GPIO_InitTypeDef       GPIO_InitStructure;
```

```
DMA_InitTypeDef        DMA_InitStructure;
ADC_InitTypeDef        ADC_InitStructure;
ADC_CommonInitTypeDef  ADC_CommonInitStructure;

/* Enable peripheral clocks *************************************************/
RCC_AHB1PeriphClockCmd( ADC1_2_CHANNEL_GPIO_CLK , ENABLE);
RCC_AHB1PeriphClockCmd( RCC_AHB1Periph_DMA2 , ENABLE);
RCC_APB2PeriphClockCmd( RCC_APB2Periph_ADC1 , ENABLE);
RCC_APB2PeriphClockCmd( RCC_APB2Periph_ADC2 , ENABLE);
RCC_APB2PeriphClockCmd( RCC_APB2Periph_ADC3 , ENABLE);

/* Configure ADC Channel 13 pin as analog input *****************************/
GPIO_InitStructure.GPIO_Pin = GPIO_PIN;
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AN;
GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL ;
GPIO_Init(GPIO_PORT, &GPIO_InitStructure);

/* DMA2 Stream0 channel0 configuration **************************************/
DMA_InitStructure.DMA_Channel = DMA_CHANNELx;
DMA_InitStructure.DMA_PeripheralBaseAddr = (uint32_t)ADC_CDR_ADDRESS;
DMA_InitStructure.DMA_Memory0BaseAddr = (uint32_t)&aADCTripleConvertedValue;
DMA_InitStructure.DMA_DIR = DMA_DIR_PeripheralToMemory;
DMA_InitStructure.DMA_BufferSize = TripleADCMessungen;
DMA_InitStructure.DMA_PeripheralInc = DMA_PeripheralInc_Disable;
DMA_InitStructure.DMA_MemoryInc = DMA_MemoryInc_Enable;
DMA_InitStructure.DMA_PeripheralDataSize = DMA_PeripheralDataSize_Word;
DMA_InitStructure.DMA_MemoryDataSize = DMA_MemoryDataSize_Word;
DMA_InitStructure.DMA_Mode = DMA_Mode_Normal;
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(DMA_STREAMx, &DMA_InitStructure);

/* DMA2_Stream0 enable */
DMA_Cmd(DMA_STREAMx, ENABLE);

/* ADC Common configuration *************************************************/
ADC_CommonInitStructure.ADC_Mode = ADC_TripleMode_Interl;
ADC_CommonInitStructure.ADC_TwoSamplingDelay = ADC_TwoSamplingDelay_5Cycles;
ADC_CommonInitStructure.ADC_DMAAccessMode = ADC_DMAAccessMode_2;
ADC_CommonInitStructure.ADC_Prescaler = ADC_Prescaler_Div2;
ADC_CommonInit(&ADC_CommonInitStructure);

/* ADC1 regular channel 12 configuration ************************************/
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_ExternalTrigConv = ADC_ExternalTrigConv_T1_CC1;
ADC_InitStructure.ADC_DataAlign = ADC_DataAlign_Right;
ADC_InitStructure.ADC_NbrOfConversion = 1;
ADC_Init(ADC1, &ADC_InitStructure);

ADC_RegularChannelConfig(ADC1, ADC_CHANNEL, 1, ADC_SampleTime_3Cycles);
/* Enable ADC1 DMA */
ADC_DMACmd(ADC1, ENABLE);

/* ADC2 regular channel 12 configuration ************************************/
ADC_Init(ADC2, &ADC_InitStructure);
/* ADC2 regular channel12 configuration */
ADC_RegularChannelConfig(ADC2, ADC_CHANNEL, 1, ADC_SampleTime_3Cycles);

/* ADC3 regular channel 12 configuration ************************************/
ADC_Init(ADC3, &ADC_InitStructure);
/* ADC3 regular channel12 configuration */
ADC_RegularChannelConfig(ADC3, ADC_CHANNEL, 1, ADC_SampleTime_3Cycles);
ADC_DMARequestAfterLastTransferCmd(ADC1, ENABLE);
```

```
  /* Enable DMA request after last transfer (multi-ADC mode) ******************/
 // ADC_MultiModeDMARequestAfterLastTransferCmd(ENABLE);

  /* Enable ADC1 ***************************************************************/
  ADC_Cmd(ADC1, ENABLE);

  /* Enable ADC2 ***************************************************************/
  ADC_Cmd(ADC2, ENABLE);

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

#ifdef  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 STMicroelectronics *****END OF FILE****/
```


4500
4000
3500
3000
2500
2000
1500
1000