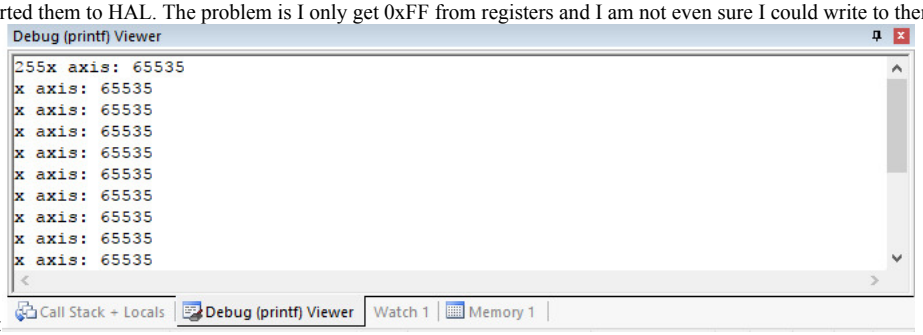
Hi, guys. I just started up with STM32F4VGT (Revision C) Discovery Board. I use HAL libraries and CubeMX to program it. I found some codes which are written in STP and I converted them to HAL. The problem is I only get 0xFF from registers and I am not even sure I could write to them. Here is the result seen on debug(printf) view:



And here is my code:

```
/**
  ******************************************************************************
  * File Name          : main.c
  * Description        : Main program body
  ******************************************************************************
  ** This notice applies to any and all portions of this file
  * that are not between comment pairs USER CODE BEGIN and
  * USER CODE END. Other portions of this file, whether
  * inserted by the user or by software development tools
  * are owned by their respective copyright owners.
  *
  * COPYRIGHT(c) 2017 STMicroelectronics
  *
  * Redistribution and use in source and binary forms, with or without modification,
  * are permitted provided that the following conditions are met:
  *   1. Redistributions of source code must retain the above copyright notice,
  *      this list of conditions and the following disclaimer.
  *   2. Redistributions in binary form must reproduce the above copyright notice,
  *      this list of conditions and the following disclaimer in the documentation
  *      and/or other materials provided with the distribution.
  *   3. Neither the name of STMicroelectronics nor the names of its contributors
  *      may be used to endorse or promote products derived from this software
  *      without specific prior written permission.
  *
  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
  * DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
  * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
  * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
  * SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
  * CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
  * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
  * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
  *
  ******************************************************************************
  */

/* Includes ------------------------------------------------------------------*/
#include "main.h"
#include "stm32f4xx_hal.h"

/* USER CODE BEGIN Includes */
/* USER CODE END Includes */

/* Private variables ---------------------------------------------------------*/
SPI_HandleTypeDef hspi1;

/* USER CODE BEGIN PV */
/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_SPI1_Init(void);

/* USER CODE BEGIN PFP */
     void SPI_send(uint8_t,uint8_t);
     uint8_t SPI_read(uint8_t);
/* USER CODE END PFP */

/* USER CODE BEGIN 0 */


/* USER CODE END 0 */

int main(void)
{

  /* USER CODE BEGIN 1 */
     volatile uint8_t spiRxBuf[2];
     volatile uint16_t x;
     volatile uint8_t WHO_AM_I;
     //char buf[10];

  /* USER CODE END 1 */

  /* MCU Configuration----------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();
```

```
  /* USER CODE BEGIN Init */
  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */
  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_SPI1_Init();

  /* USER CODE BEGIN 2 */
     //HAL_GPIO_WritePin(GPIOE,GPIO_PIN_3,GPIO_PIN_SET);
     SPI_send(0x23,0xc9);
     SPI_send(0x20,0x17);
     SPI_send(0x24,0x20);
     SPI_send(0x10,0x00);
     SPI_send(0x11,0x00);
     SPI_send(0x12,0x00);

     WHO_AM_I = SPI_read(0x0F);
     //sprintf(buf,"%d",WHO_AM_I);
     printf("%d",WHO_AM_I);

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
     while(1)
     {

          spiRxBuf[0] = SPI_read(0x28);

          spiRxBuf[1] = SPI_read(0x29);

          x = (spiRxBuf[1] << 8) | spiRxBuf[0];
          //sprintf(bufx,"%d",spiRxBuf[0]);



          //sprintf(buf,"%d",x);

          printf("x axis: %d",x);
          printf("\n");
          HAL_Delay(1000);
     }

  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
  /* USER CODE END 3 */

}

/** System Clock Configuration
*/
void SystemClock_Config(void)
{

  RCC_OscInitTypeDef RCC_OscInitStruct;
  RCC_ClkInitTypeDef RCC_ClkInitStruct;

    /**Configure the main internal regulator output voltage
    */
  __HAL_RCC_PWR_CLK_ENABLE();

  __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);

    /**Initializes the CPU, AHB and APB busses clocks
    */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSICalibrationValue = 16;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_NONE;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

    /**Initializes the CPU, AHB and APB busses clocks
    */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_HSI;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_0) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

    /**Configure the Systick interrupt time
    */
  HAL_SYSTICK_Config(HAL_RCC_GetHCLKFreq()/1000);

    /**Configure the Systick
    */
  HAL_SYSTICK_CLKSourceConfig(SYSTICK_CLKSOURCE_HCLK);

  /* SysTick_IRQn interrupt configuration */
```

```
  HAL_NVIC_SetPriority(SysTick_IRQn, 0, 0);
}

/* SPI1 init function */
static void MX_SPI1_Init(void)
{

  /* SPI1 parameter configuration*/
  hspi1.Instance = SPI1;
  hspi1.Init.Mode = SPI_MODE_MASTER;
  hspi1.Init.Direction = SPI_DIRECTION_2LINES;
  hspi1.Init.DataSize = SPI_DATASIZE_8BIT;
  hspi1.Init.CLKPolarity = SPI_POLARITY_HIGH;
  hspi1.Init.CLKPhase = SPI_PHASE_2EDGE;
  hspi1.Init.NSS = SPI_NSS_SOFT;
  hspi1.Init.BaudRatePrescaler = SPI_BAUDRATEPRESCALER_256;
  hspi1.Init.FirstBit = SPI_FIRSTBIT_MSB;
  hspi1.Init.TIMode = SPI_TIMODE_DISABLE;
  hspi1.Init.CRCCalculation = SPI_CRCCALCULATION_DISABLE;
  hspi1.Init.CRCPolynomial = 10;
  if (HAL_SPI_Init(&hspi1) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

}

/** Configure pins as
        * Analog
        * Input
        * Output
        * EVENT_OUT
        * EXTI
*/
static void MX_GPIO_Init(void)
{

  GPIO_InitTypeDef GPIO_InitStruct;

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOE_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOD_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOE, GPIO_PIN_3, GPIO_PIN_SET);

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_RESET);

  /*Configure GPIO pin : PE3 */
  GPIO_InitStruct.Pin = GPIO_PIN_3;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
  HAL_GPIO_Init(GPIOE, &GPIO_InitStruct);

  /*Configure GPIO pin : PD13 */
  GPIO_InitStruct.Pin = GPIO_PIN_13;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOD, &GPIO_InitStruct);

}

/* USER CODE BEGIN 4 */

void SPI_send(uint8_t address,uint8_t data)
{

          uint8_t rx[2];

          while(HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY)
          {};

          HAL_GPIO_WritePin(GPIOE,GPIO_PIN_3,GPIO_PIN_RESET);

          HAL_SPI_Transmit(&hspi1,&address,1,HAL_MAX_DELAY);

          while(HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY)
          {};

          HAL_SPI_Receive(&hspi1,rx,1,HAL_MAX_DELAY);


               while(HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY)
          {};

          HAL_SPI_Transmit(&hspi1,&data,1,HAL_MAX_DELAY);

               while(HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY)
          {};

          HAL_SPI_Receive(&hspi1,rx,1,HAL_MAX_DELAY);

               while(HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY)
          {};

          HAL_GPIO_WritePin(GPIOE,GPIO_PIN_3,GPIO_PIN_SET);

}
```

```
uint8_t SPI_read(uint8_t address)
{
     uint8_t rx,garbage;
     address = address | 0x80;

     while(HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY)
          {};

     HAL_GPIO_WritePin(GPIOE,GPIO_PIN_3,GPIO_PIN_RESET);

     HAL_SPI_Transmit(&hspi1,&address,1,HAL_MAX_DELAY);

               while(HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY)
          {};

     HAL_SPI_Receive(&hspi1,&rx,1,HAL_MAX_DELAY);

               while(HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY)
          {};

     HAL_SPI_Transmit(&hspi1,0x00,1,HAL_MAX_DELAY);

               while(HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY)
          {};

     HAL_SPI_Receive(&hspi1,&garbage,1,HAL_MAX_DELAY);

               while(HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY)
          {};

     HAL_GPIO_WritePin(GPIOE,GPIO_PIN_3,GPIO_PIN_SET);

     return rx;

}
/* USER CODE END 4 */

/**
  * @brief  This function is executed in case of error occurrence.
  * @param  None
  * @retval None
  */
void _Error_Handler(char * file, int line)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* USER CODE END Error_Handler_Debug */
}

#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 CODE BEGIN 6 */
  /* USER CODE END 6 */

}

#endif

/**
  * @}
  */

/**
  * @}
*/

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