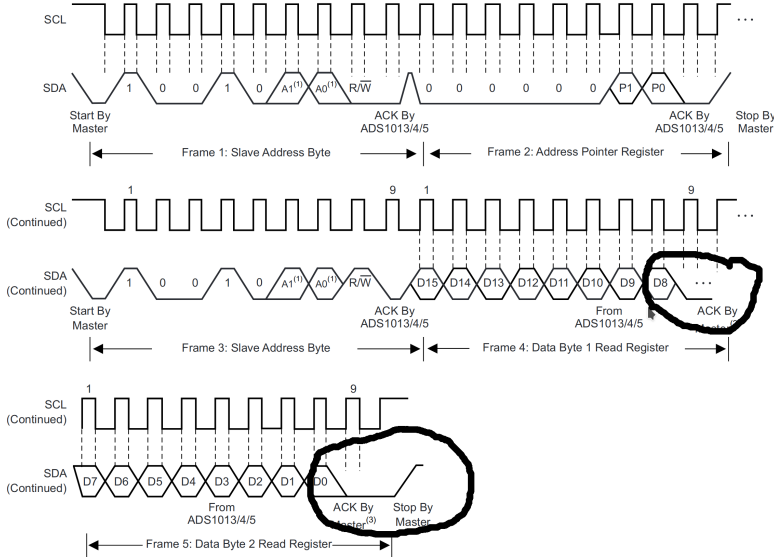
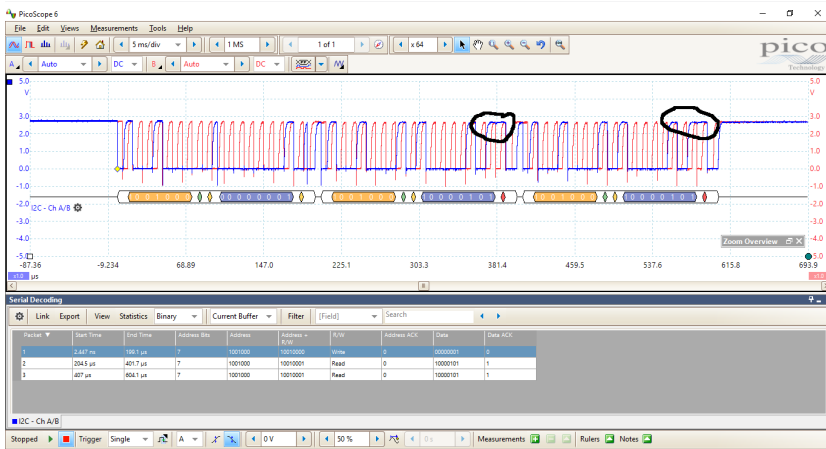


I am trying to interface with an ADS1015 using an STM32F103. I am having problems reading two bytes from the ADS1015. The first byte is received perfectly, but the second byte is exactly the same as the first byte which is incorrect. I think this is because the master (STM32) is not acknowledging the bytes being received. This can be seen below. This is what should be seen according to the datasheet.



However instead this is seen



You can see that the master should be pulling the line low at these points (Acknowledging) but it is not doing this. Here is my code

```
#include "stm32f1xx.h"
#include "stm32f1xx_hal.h"
#include "stm32f1xx_hal_conf.h"

#define I2C_SPEEDCLOCK 100000

I2C_HandleTypeDef I2cHandle;
UART_HandleTypeDef UartHandle;

__IO ITStatus UartReady = RESET;

void SystemClock_Config(void);
void Startup_Sequence(void);
void Error_Handler(void);
void i2c1_send_data(uint16_t addr, uint8_t* data, uint8_t length);
void i2c1_receive_data(uint16_t addr, uint8_t* data, uint8_t length);
static void i2cUnstick(void);

/**
 * This function is called when transmitting
 * @param UartHandle [Pointer to UartHandle]
 */
void HAL_UART_TxCpltCallback(UART_HandleTypeDef *UartHandle) {
    UartReady = SET;
}

/**
 * This function is called when receiving
 * @param UartHandle [Pointer to UartHandle]
 */
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *UartHandle) {
    UartReady = SET;
}

/**
 * Associates the interrupt handler with the UartHandle
 */
void USART1_IRQHandler(void) {
    HAL_UART_IRQHandler(&UartHandle);
}

int main(void)
{
    HAL_Init();

```

```

SystemInit();
SystemClock_Config();

GPIO_InitTypeDef GPIO_InitStructure;
__GPIO_CLK_ENABLE();
__GPIOB_CLK_ENABLE();
__GPIOA_CLK_ENABLE();
__HAL_RCC_I2C1_CLK_ENABLE();
__HAL_RCC_USART1_CLK_ENABLE();

//Configuring pin for onboard LED.
GPIO_InitStructure.Pin = GPIO_PIN_13;
GPIO_InitStructure.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStructure.Pull = GPIO_NOPULL;
GPIO_InitStructure.Speed = GPIO_SPEED_HIGH;
HAL_GPIO_Init(GPIOC, &GPIO_InitStructure);

//Configuring SDA and SCL pins for I2C.
GPIO_InitStructure.Pin = GPIO_PIN_6 | GPIO_PIN_7;
GPIO_InitStructure.Mode = GPIO_MODE_AF_OD;
HAL_GPIO_Init(GPIOB, &GPIO_InitStructure);

// Setup UART TX Pin
GPIO_InitStructure.Pin = GPIO_PIN_9;
GPIO_InitStructure.Mode = GPIO_MODE_AF_PP;
GPIO_InitStructure.Pull = GPIO_PULLUP;
GPIO_InitStructure.Speed = GPIO_SPEED_HIGH;
HAL_GPIO_Init(GPIOA, &GPIO_InitStructure);

//Setup UART RX Pin
GPIO_InitStructure.Pin = GPIO_PIN_10;
GPIO_InitStructure.Mode = GPIO_MODE_INPUT;
GPIO_InitStructure.Speed = GPIO_SPEED_HIGH;
HAL_GPIO_Init(GPIOA, &GPIO_InitStructure);

UartHandle.Instance = USART1;
UartHandle.Init.BaudRate = 9600;
UartHandle.Init.WordLength = UART_WORDLENGTH_8B;
UartHandle.Init.StopBits = UART_STOPBITS_1;
UartHandle.Init.Parity = UART_PARITY_NONE;
UartHandle.Init.HwFlowCtl = UART_HWCONTROL_NONE;
UartHandle.Init.Mode = UART_MODE_TX_RX;

if(HAL_UART_Init(&UartHandle) != HAL_OK) {
    Error_Handler();
}

HAL_NVIC_SetPriority(USART1_IRQn, 0, 1);
HAL_NVIC_EnableIRQ(USART1_IRQn);

I2cHandle.Instance = I2C1;
I2cHandle.Init.ClockSpeed = I2C_SPEEDCLOCK;
I2cHandle.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;

HAL_NVIC_SetPriority(I2C1_ER_IRQn, 0, 1);
HAL_NVIC_EnableIRQ(I2C1_ER_IRQn);
HAL_NVIC_SetPriority(I2C1_EV_IRQn, 0, 1);
HAL_NVIC_EnableIRQ(I2C1_EV_IRQn);

if (HAL_I2C_Init(&I2cHandle) != HAL_OK) {
    Error_Handler();
}

Startup_Sequence();
uint16_t config;
uint8_t configRegister = 1;

i2c1_send_data(0b10010000, &configRegister, 1);
i2c1_receive_data(0b10010000, &config, 1);
i2c1_receive_data(0b10010000, &config, 1);
//HAL_UART_Transmit_IT(&UartHandle, &M5B, 1);
//HAL_UART_Transmit_IT(&UartHandle, &config, 1);
while (1) {
}
}

/**
 * Configures System Clock. Configured as follows:
 * System Clock source = PLL (HSE)
 * SYSCLK (Hz) = 72000000
 * HCLK (Hz) = 72000000
 * AHB Prescaler = 1
 * APB1 Prescaler = 2
 * APB2 Prescaler = 1
 * HSE Frequency (Hz) = 80000000
 * HSE PREDIV1 = 1
 * PLLMUL = 9
 * Flash Latency (WS) = 2
 */
void SystemClock_Config(void) {
    RCC_ClkInitTypeDef clkinitstruct = {0};
    RCC_OscInitTypeDef oscinitstruct = {0};
    oscinitstruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
    oscinitstruct.HSEState = RCC_HSE_ON;
    oscinitstruct.HSEPredivValue = RCC_HSE_PREDIV_DIV1;
    oscinitstruct.PLL.PLLState = RCC_PLL_ON;
    oscinitstruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
    oscinitstruct.PLL.PLLMUL = RCC_PLL_MUL9;
    if (HAL_RCC_OscConfig(&oscinitstruct) != HAL_OK) {
        while(1);
    }
    clkinitstruct.ClockType = (RCC_CLOCKTYPE_SYSCLK | RCC_CLOCKTYPE_HCLK | RCC_CLOCKTYPE_PCLK1 | RCC_CLOCKTYPE_PCLK2);
    clkinitstruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;

```

