Hello,

I've been working with ST HAL libraries for an STM32F030R8T6, using the NUCLEO board, and I'm having some problems with the I2C API. Probably it's a configuration issue but for the love of me, I can't find it.

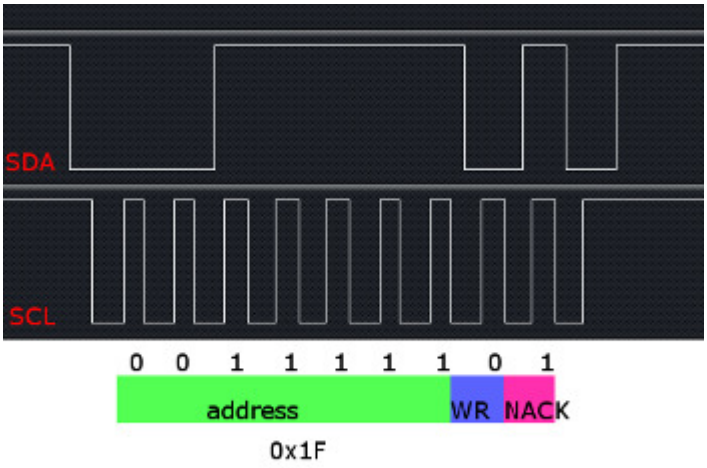
The problem:
I call the function HAL_I2C_Master_Transmit_XX with address 0x3E. I attach a logic analyzer to the output to check the lines, and the address does not match. The address I see on the line is 0x1F.

My main test code is the following. (Just to reproduce this situation) It is a direct copy from the NUCLEO examples.
I skip over the SystemClock_Config and MspInit, since I can see data on the lines (I2C is sending data correcly) and they don't provide information for my current problem.

```
01. #define I2C_ADDRESS        0x3E
02. #define I2C_TIMING      0x00A51314
03. uint8_t bTransferRequest = 0;
04.
05. //(other stuff)
06.
07. int main(void)
08. {
09.   HAL_Init();
10.
11.   /* Configure the system clock to 48 MHz */
12.   SystemClock_Config();
13.
14.   /*##-1- Configure the I2C peripheral ######################################*/
15.   I2cHandle.Instance             = I2C1;
16.
17.   I2cHandle.Init.Timing                     = I2C_TIMING;
18.   I2cHandle.Init.OwnAddress1           = I2C_ADDRESS;
19.   I2cHandle.Init.AddressingMode     = I2C_ADDRESSINGMODE_7BIT;
20.   I2cHandle.Init.DualAddressMode   = I2C_DUALADDRESS_DISABLE;
21.   I2cHandle.Init.OwnAddress2           = 0xFF;
22.   I2cHandle.Init.GeneralCallMode     = I2C_GENERALCALL_DISABLE;
23.   I2cHandle.Init.NoStretchMode        = I2C_NOSTRETCH_DISABLE;
24.
25.   if(HAL_I2C_Init(&I2cHandle) != HAL_OK)
26.   {
27.     /* Initialization Error */
28.     Error_Handler();
29.   }
30.
31.   /* Enable the Analog I2C Filter */
32.   HAL_I2CEx_ConfigAnalogFilter(&I2cHandle,I2C_ANALOGFILTER_ENABLE);
33.


34.
35.   while(1){
36.       HAL_I2C_Master_Transmit(&I2cHandle, (uint16_t)I2C_ADDRESS,
(uint8_t*)&bTransferRequest, 1, 1000);
37.       HAL_Delay(1000);
38.   }
39.
40. return 0;
41. }
```

This code, when examining the output with the logic analyzer, generates the following SDA/SCL sequence.



The Theory:
I went down to the ST code generating the address.

```
/**********************************************************************
  * @file    stm32f0xx_hal_i2c.c
  * @author  MCD Application Team
  * @version V1.3.0
  * @date    26-June-2015
 **********************************************************************/
//(etc)

static void I2C_TransferConfig(I2C_HandleTypeDef *hi2c,  uint16_t DevAddress, uint8_t Size,
uint32_t Mode, uint32_t Request){
//(etc)
  /* update tmpreg */
  tmpreg |= (uint32_t)(((uint32_t)DevAddress & I2C_CR2_SADD) | (((uint32_t)Size << 16 ) &
I2C_CR2_NBYTES) | \
             (uint32_t)Mode | (uint32_t)Request);

  /* update CR2 register */
  hi2c->Instance->CR2 = tmpreg;
```

And from the reference manual for STM32F0 devices, section 25.7.2, Control Register 2 (I2C_CR2):

```
Bits 9:8 SADD[9:8]: Slave address bit 9:8 (master mode)
    This bit are don?t care
Bits 7:1 SADD[7:1]: Slave address bit 7:1 (master mode)
    These bits should be written with the 7-bit slave address to be sent
Bit 0 SADD0: Slave address bit 0 (master mode)
    This bit is don?t care
```

I understand that on 7 bit address mode, the CR2 bits used to generate the address are SADD[7:1] neglecting the value of SADD0. But the ST HAL driver seems to write the value directly (with no shifting at all), hence the "transformation" from 0x3E to 0x1F:
I2C_CR2[7:0] = 0x3E
I2C_CR2[7:1] = 0x1F
This would be fine for 10 bit addresses, but I'm having trouble with 7 bit.

I tested using an I2C temperature sensor (address 0x40). And I can't read anything at all using 0x40, but everything is fine and dandy when using 0x80.

Could somebody please point me to any configuration flag that I may have missed?

Salut!