**#include** "stm32f4xx.h"

**#include** "stm32f429i\_discovery.h"

**#include** "stdint.h"

**#include** "stm32f429i\_discovery.h"

**#include** "stm32f4xx\_spi.h"

**#include** "defines.h"

**#include** "tm\_stm32f4\_ili9341.h"

**#include** "tm\_stm32f4\_fonts.h"

**#include** "i3g4250d.h"

**#include** <stdio.h>

**#include** <string.h>

**#include** <stdlib.h>

**#define** FILTER\_SIZE 10

int16\_t x\_values[FILTER\_SIZE] = {0};

int16\_t y\_values[FILTER\_SIZE] = {0};

int16\_t z\_values[FILTER\_SIZE] = {0};

GPIO\_InitTypeDef GPIO\_InitStruct;

SPI\_InitTypeDef SPI\_InitStruct;

**int** x, y, z;

uint8\_t x\_address = 0x29;

uint8\_t y\_address = 0x2B;

uint8\_t z\_address = 0x2D;

uint8\_t my\_data = 0x00;

**void** **GPIO\_Config**()

{

RCC\_AHB1PeriphClockCmd(RCC\_AHB1Periph\_GPIOG, *ENABLE*);

RCC\_AHB1PeriphClockCmd(RCC\_AHB1Periph\_GPIOC, *ENABLE*);

RCC\_AHB1PeriphClockCmd(RCC\_AHB1Periph\_GPIOF, *ENABLE*);

//RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_SPI5, ENABLE);

GPIO\_InitStruct.GPIO\_Pin = GPIO\_Pin\_13 | GPIO\_Pin\_14;

GPIO\_InitStruct.GPIO\_Mode = *GPIO\_Mode\_OUT*;

GPIO\_InitStruct.GPIO\_OType = *GPIO\_OType\_PP*;

GPIO\_InitStruct.GPIO\_PuPd = *GPIO\_PuPd\_NOPULL*;

GPIO\_InitStruct.GPIO\_Speed = *GPIO\_Speed\_100MHz*;

GPIO\_Init(GPIOG, &GPIO\_InitStruct);

GPIO\_InitStruct.GPIO\_Mode=*GPIO\_Mode\_OUT*;

GPIO\_InitStruct.GPIO\_Pin=GPIO\_Pin\_1;

GPIO\_InitStruct.GPIO\_OType=*GPIO\_OType\_PP*;

GPIO\_InitStruct.GPIO\_PuPd = *GPIO\_PuPd\_NOPULL*;

GPIO\_InitStruct.GPIO\_Speed = *GPIO\_Speed\_100MHz*;

GPIO\_Init(GPIOC, &GPIO\_InitStruct);

GPIO\_PinAFConfig(GPIOF, GPIO\_PinSource7, GPIO\_AF\_SPI5);

GPIO\_PinAFConfig(GPIOF, GPIO\_PinSource8, GPIO\_AF\_SPI5);

GPIO\_PinAFConfig(GPIOF, GPIO\_PinSource9, GPIO\_AF\_SPI5);

GPIO\_InitStruct.GPIO\_Mode = *GPIO\_Mode\_AF*;

GPIO\_InitStruct.GPIO\_Pin = GPIO\_Pin\_7 | GPIO\_Pin\_8 | GPIO\_Pin\_9;

GPIO\_InitStruct.GPIO\_OType = *GPIO\_OType\_PP*;

GPIO\_InitStruct.GPIO\_PuPd = *GPIO\_PuPd\_NOPULL*;

GPIO\_InitStruct.GPIO\_Speed = *GPIO\_Speed\_100MHz*;

GPIO\_Init(GPIOF, &GPIO\_InitStruct);

}

**void** **SPI\_Config**()

{

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_SPI5, *ENABLE*);

SPI\_InitStruct.SPI\_BaudRatePrescaler = SPI\_BaudRatePrescaler\_2;

SPI\_InitStruct.SPI\_CPHA = SPI\_CPHA\_2Edge;

SPI\_InitStruct.SPI\_CPOL = SPI\_CPOL\_High;

SPI\_InitStruct.SPI\_DataSize = SPI\_DataSize\_8b;

SPI\_InitStruct.SPI\_Direction = SPI\_Direction\_2Lines\_FullDuplex;

SPI\_InitStruct.SPI\_FirstBit = SPI\_FirstBit\_MSB;

SPI\_InitStruct.SPI\_Mode = SPI\_Mode\_Master;

SPI\_InitStruct.SPI\_NSS = SPI\_NSS\_Soft | SPI\_NSSInternalSoft\_Set;

SPI\_Init(SPI5, &SPI\_InitStruct);

SPI\_Cmd(SPI5, *ENABLE*);

GPIO\_SetBits(GPIOC, GPIO\_Pin\_1);

}

**void** **SPI\_Write**(uint8\_t address, uint8\_t data)

{

GPIO\_ResetBits(GPIOC, GPIO\_Pin\_1);

**while** (!SPI\_I2S\_GetFlagStatus(SPI5, SPI\_I2S\_FLAG\_TXE));

SPI\_I2S\_SendData(SPI5, address);

**while** (!SPI\_I2S\_GetFlagStatus(SPI5, SPI\_I2S\_FLAG\_RXNE));

SPI\_I2S\_ReceiveData(SPI5);

**while** (!SPI\_I2S\_GetFlagStatus(SPI5, SPI\_I2S\_FLAG\_TXE));

SPI\_I2S\_SendData(SPI5, data);

**while** (!SPI\_I2S\_GetFlagStatus(SPI5, SPI\_I2S\_FLAG\_RXNE));

SPI\_I2S\_ReceiveData(SPI5);

GPIO\_SetBits(GPIOC, GPIO\_Pin\_1);

}

uint8\_t **SPI\_Read**(uint8\_t address,uint8\_t data)

{

GPIO\_ResetBits(GPIOC, GPIO\_Pin\_1);

address = address | 0x80;

**while** (!SPI\_I2S\_GetFlagStatus(SPI5, SPI\_I2S\_FLAG\_TXE));

SPI\_I2S\_SendData(SPI5, address);

**while** (!SPI\_I2S\_GetFlagStatus(SPI5, SPI\_I2S\_FLAG\_RXNE));

SPI\_I2S\_ReceiveData(SPI5);

**while** (!SPI\_I2S\_GetFlagStatus(SPI5, SPI\_I2S\_FLAG\_TXE));

SPI\_I2S\_SendData(SPI5, data);

**while** (!SPI\_I2S\_GetFlagStatus(SPI5, SPI\_I2S\_FLAG\_RXNE));

SPI\_I2S\_ReceiveData(SPI5);

GPIO\_SetBits(GPIOC, GPIO\_Pin\_1);

**return** SPI\_I2S\_ReceiveData(SPI5);

}

**int** **main**(**void**)

{

GPIO\_Config();

SPI\_Config();

SPI\_Write(0x20, 0x0f);

SystemInit();

//Initialize ILI9341

TM\_ILI9341\_Init();

//Rotate LCD for 90 degrees

TM\_ILI9341\_Rotate(*TM\_ILI9341\_Orientation\_Landscape\_2*);

//FIll lcd with color

TM\_ILI9341\_Fill(ILI9341\_COLOR\_GREEN);

**char** buffer [33];

**while** (1)

{

x = SPI\_Read(x\_address, SPI\_Read(x\_address, x));

y = SPI\_Read(y\_address, SPI\_Read(y\_address, y));

z = SPI\_Read(z\_address, SPI\_Read(z\_address, z));

TM\_ILI9341\_Puts(125, 10, **itoa**(x,buffer,10), &TM\_Font\_16x26, ILI9341\_COLOR\_BLACK, ILI9341\_COLOR\_ORANGE);

TM\_ILI9341\_Puts(125, 50, **itoa**(y,buffer,10), &TM\_Font\_16x26, ILI9341\_COLOR\_BLACK, ILI9341\_COLOR\_ORANGE);

TM\_ILI9341\_Puts(125, 100, **itoa**(z,buffer,10), &TM\_Font\_16x26, ILI9341\_COLOR\_BLACK, ILI9341\_COLOR\_ORANGE);

**for** (**int** i = 0; i<720000; i++);

}

}

uint32\_t **sEE\_TIMEOUT\_UserCallback**(**void**)

{

/\* **TODO**, implement your code here \*/

**while** (1)

{

}

}

**void** **EVAL\_AUDIO\_TransferComplete\_CallBack**(uint32\_t pBuffer, uint32\_t Size)

{

**return**;

}

uint16\_t **EVAL\_AUDIO\_GetSampleCallBack**(**void**)

{

**return** -1;

}