#include <stm32f10x.h>//definición de los nombres

int x;

int y;

int z;

int out;

//THERE SHOULD BE A DELAY BETWEEN TIMER ON AND START OF CONVERSION

//check for a converter channel not in the connected range?

//resolution (12 bits) (Vref + aprox 3 Volts) (Vref- = 0 I suppose)

int main(void)

{

out = 0;

RCC->APB2ENR |= (1<<9)| (1<<2) | (1<<4) ;//Reset and clock control: Del registro RCC se accede a la parte APB2ENR

GPIOA->CRL = 0; //PUERTO A, que tiene el ADC, configurado como INPUT, ANALOGO

GPIOC->CRL = 0x88888888;//PUERTO C LOW, config en general purpose output, max speed 2MHz

//GPIOC->CRH = 0x88888888;//PUERTO C HIGH, config en general purpose output, max speed 2MHz

NVIC\_EnableIRQ(ADC1\_IRQn);

ADC1->CR1 = (1<<5); //habilita EOC, que nos permite saber que acabo la conversion

ADC1->SMPR2 = 0x70; //sample time for each channel set to 239.5 cycles (nou enteeandou)/

ADC1->CR2 = 0x1; //ADC ON , single mode(cont=0),right alignment (bit 11)

ADC1->SQR1 = 0; // total number of conversions (one conversion is a 0 in the bits #20-23), other bits are order of the conversion of channel, from last to first going down

ADC1->SQR2 = 0; // no channels converted in this range (from 12th to 7th)

GPIOC->ODR = out;

while(1)

{

//ADC1->SQR3 = 1; // in the first 5 bits you put the channel # of the channel you want to be converted first, in this case channel 1 (second in position)

//ADC1->CR2 |= ADC\_CR2\_ADON; //start conversion (flanco?)

ADC1->SQR3 = 2; // in the first 5 bits you put the channel # of the channel you want to be converted first, in this case channel 2 (second in position)

ADC1->CR2 |= ADC\_CR2\_ADON; //start conversion (flanco?)

//ADC1->SQR3 = 3; // in the first 5 bits you put the channel # of the channel you want to be converted first, in this case channel 3 (second in position)

//ADC1->CR2 |= ADC\_CR2\_ADON; //start conversion (flanco?)

if (y > 3072)

{

//out = 1;

}

/\* if (y < 1024)

{

out = 2;

}

if (x > 3072)

{

out = 4;

}

if (x < 1024)

{

out = 8;

}

if ((y > 3072) & (x > 3072))

{

out = 5;

}

if ((y > 3072) & (x < 1024))

{

out = 9;

}

if ((y < 1024) & (x > 3072))

{

out = 6;

}

if ((y < 1024) & (x < 1024))

{

out = 10;

}

\*/

GPIOC->ODR = out;

}

}

void ADC1\_IRQHandler() // (?) (interrupcion)

{

if((ADC1->SR & ADC\_SR\_EOC) & (ADC1->SQR3 = 1)) //hace una AND con en bit 1(posicion 2) del registro de estado a ver si ya acabo la conversion, y aunque el resultado no valdria exactamente 1, sino dos, supongo que if lo toma como true.. por lo que supongo que seria: 0 = no se da la condicion. todo lo demas= se da la condecion)

{

x = ADC1->DR; //CONVERSION put on x

//GPIOC->ODR = ADC1->DR; //CONVERSION put on PORT C ODR (output data register)

ADC1->SR = 0; //Status register must be cleared by software

}

if((ADC1->SR & ADC\_SR\_EOC) & (ADC1->SQR3 = 2)) //hace una AND con en bit 1(posicion 2) del registro de estado a ver si ya acabo la conversion, y aunque el resultado no valdria exactamente 1, sino dos, supongo que if lo toma como true.. por lo que supongo que seria: 0 = no se da la condicion. todo lo demas= se da la condecion)

{

y = ADC1->DR; //CONVERSION put on y

//GPIOC->ODR = ADC1->DR; //CONVERSION put on PORT C ODR (output data register)

ADC1->SR = 0; //Status register must be cleared by software

}

if((ADC1->SR & ADC\_SR\_EOC) & (ADC1->SQR3 = 3)) //hace una AND con en bit 1(posicion 2) del registro de estado a ver si ya acabo la conversion, y aunque el resultado no valdria exactamente 1, sino dos, supongo que if lo toma como true.. por lo que supongo que seria: 0 = no se da la condicion. todo lo demas= se da la condecion)

{

z = ADC1->DR; //CONVERSION put on z

//GPIOC->ODR = ADC1->DR; //CONVERSION put on PORT C ODR (output data register)

ADC1->SR = 0; //Status register must be cleared by software

}

ADC1->SR = 0; //Status register must be cleared by software

}