#include "mbed.h"

#define rise\_edge 0

#define fall\_edge 1

void timer\_meas(bool fall\_rise);

class Counter {

public:

Counter(PinName pin) : \_interrupt(pin) { // create the InterruptIn on the pin specified to Counter

\_interrupt.rise(this, &Counter::rising); // attach increment function of this counter instance

\_interrupt.fall(this, &Counter::falling); // attach increment function of this counter instance

}

void rising() {

timer\_meas(rise\_edge);

}

void falling() {

timer\_meas(fall\_edge);

}

int read() {

return \_count;

}

private:

InterruptIn \_interrupt;

volatile int \_count;

};

Counter counter(A5);

Timer AM;

struct frame {

bool SOF;

int timer\_val;

int last\_timer;

};

struct frame frame\_byte;

bool timer\_running=false;

void timer\_meas(bool fall\_rise) {

if(fall\_rise == fall\_edge&&timer\_running ==false){

AM.reset();

timer\_running = true;

} else if(frame\_byte.SOF == false){

frame\_byte.timer\_val=AM.read\_us();

switch (fall\_rise) {

case fall\_edge:

if(frame\_byte.last\_timer == 10&&((40<= frame\_byte.timer\_val)&&(frame\_byte.timer\_val <=70))){

frame\_byte.SOF=true;

} else{

timer\_running=false;

}

break;

case rise\_edge:

if(frame\_byte.last\_timer == 0&&((5<= frame\_byte.timer\_val)&&(frame\_byte.timer\_val <=15))){

frame\_byte.last\_timer = 10;

}else{

timer\_running=false;

}

break;

default:

timer\_running=false;

break;

}

}

}

AnalogIn analog\_value\_0(A0);

AnalogIn analog\_value\_1(A1);

AnalogIn analog\_value\_2(A2);

DigitalOut led2(LED2);

DigitalOut led1(LED1);

int main() {

double x1[4];

double x2[4];

double x3[4];

double meas\_0, meas\_1, meas\_2, sumx1\_2, sumx1\_10, sumx1\_20, sumx1\_40, sumx2\_2, sumx2\_10, sumx2\_20, sumx2\_40, sumx3\_2, sumx3\_10, sumx3\_20, sumx3\_40, x1\_2, x2\_2, x3\_2, x1\_10, x2\_10, x3\_10,x1\_20, x2\_20, x3\_20,x1\_40, x2\_40, x3\_40;

int i1, i2, i3, i4;

//char c1 = in.getc();

// printf("c1: %c \n", c1);

AM.start();

frame\_byte.SOF = false;

timer\_running=false;

while(1){

//if (c1 == 's')

if (frame\_byte.SOF == true)

{ led1 = 1;

wait (1);

for (i1 = 1; i1 <=10; i1++){

wait\_ms(10);

meas\_0 = analog\_value\_0.read(); // Converts and read the analog input value (value from 0.0 to 1.0)

x1[i1] = meas\_0 \* 3.3; // Change the value to be in the 0 to 3300 range

sumx1\_2 += x1[i1];

meas\_1 = analog\_value\_1.read(); // Converts and read the analog input value (value from 0.0 to 1.0)

x2[i1] = meas\_1 \* 3.3; // Change the value to be in the 0 to 3300 range

sumx2\_2 += x2[i1];

meas\_2 = analog\_value\_2.read(); // Converts and read the analog input value (value from 0.0 to 1.0)

x3[i1] = meas\_2 \* 3.3; // Change the value to be in the 0 to 3300 range

sumx3\_2 += x3[i1];

}

x1\_2 = sumx1\_2/10;

printf ("%f, ",x1\_2) ;

x2\_2 = sumx1\_2/10;

printf ("%f, ",x2\_2);

x3\_2 = sumx1\_2/10;

printf ("%f, ",x3\_2);

led1 = 0;

wait(4);

led1 = 1;

for (i2 = 1; i2 <= 10; i3++){

meas\_0 = analog\_value\_0.read(); // Converts and read the analog input value (value from 0.0 to 1.0)

x1[i2] = meas\_0 \* 3.3; // Change the value to be in the 0 to 3300 range

sumx1\_10 += x1[i2];

meas\_1 = analog\_value\_1.read(); // Converts and read the analog input value (value from 0.0 to 1.0)

x2[i2] = meas\_1 \* 3.3; // Change the value to be in the 0 to 3300 range

sumx2\_10 += x2[i2];

meas\_2 = analog\_value\_2.read(); // Converts and read the analog input value (value from 0.0 to 1.0)

x3[i2] = meas\_2 \* 3.3; // Change the value to be in the 0 to 3300 range

sumx3\_10 += x3[i2];

}

x1\_10 = sumx1\_10/10;

printf ("%f, ",x1\_10) ;

x2\_10 = sumx2\_10/10;

printf ("%f, ",x2\_10);

x3\_10 = sumx3\_2/10;

printf ("%f, ",x3\_10);

led1 = 0;

wait(4);

led1 = 1;

for (i3 = 1; i3 <=10; i3++){

meas\_0 = analog\_value\_0.read(); // Converts and read the analog input value (value from 0.0 to 1.0)

x1[i3] = meas\_0 \* 3.3; // Change the value to be in the 0 to 3300 range

sumx1\_20 += x1[i3];

meas\_1 = analog\_value\_1.read(); // Converts and read the analog input value (value from 0.0 to 1.0)

x2[i3] = meas\_1 \* 3.3; // Change the value to be in the 0 to 3300 range

sumx2\_20 += x2[i3];

meas\_2 = analog\_value\_2.read(); // Converts and read the analog input value (value from 0.0 to 1.0)

x3[i3] = meas\_2 \* 3.3; // Change the value to be in the 0 to 3300 range

sumx3\_20 += x3[i3];

}

x1\_20 = sumx1\_20/10;

printf ("%f, ",x1\_20) ;

x2\_20 = sumx2\_20/10;

printf ("%f, ",x2\_20);

x3\_20 = sumx3\_20/10;

printf ("%f, ",x3\_20);

led1 = 0;

wait(4);

led1 = 1;

for (i4 = 1; i4 <=10; i4++){

meas\_0 = analog\_value\_0.read(); // Converts and read the analog input value (value from 0.0 to 1.0)

x1[i4] = meas\_0 \* 3.3; // Change the value to be in the 0 to 3300 range

sumx1\_40 += x1[i4];

meas\_1 = analog\_value\_1.read(); // Converts and read the analog input value (value from 0.0 to 1.0)

x2[i4] = meas\_1 \* 3.3; // Change the value to be in the 0 to 3300 range

sumx2\_40 += x2[i4];

meas\_2 = analog\_value\_2.read(); // Converts and read the analog input value (value from 0.0 to 1.0)

x3[i4] = meas\_2 \* 3.3; // Change the value to be in the 0 to 3300 range

sumx3\_40 += x3[i4];

}

x1\_40 = sumx1\_40/10;

printf ("%f, ",x1\_40) ;

x2\_40 = sumx2\_40/10;

printf ("%f, ",x2\_40);

x3\_40 = sumx3\_40/10;

printf ("%f, ",x3\_40);

led1 = 0;

timer\_running = false;

frame\_byte.SOF = false;

frame\_byte.last\_timer = 0;

frame\_byte.timer\_val = 0;

}

}

}