CODE EMETTEUR

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Test MOD-24LR / nrf24L01 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* LEnsE / Julien VILLEMEJANE / Institut d'Optique Graduate School \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Brochage \*/

/\* TO COMPLETE \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Test réalisé sur Nucléo-F411RE \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "mbed.h"

#include "MOD24\_NRF.h"

#include "nRF24.h"

Serial debug\_pc(USBTX, USBRX);

InterruptIn bp\_int(USER\_BUTTON);

char tab[8] = {0xAA, 0x01, 0x10, 0xF0,0xAA, 0x01, 0x10, 0xF0};

AnalogIn analog\_in(A5);

AnalogIn analog\_in2(A1);

/\* Main Function \*/

int main() {

double meas;

double tensionUD;

double meas\_LR;

double tension\_LR;

double varUD;

double varLR;

debug\_pc.printf("Test\r\n");

initNRF24();

while(1) {

meas = analog\_in.read(); // Convertit et lit la tension d'entée analogique (valeur entre 0 et 1)

tensionUD = meas \* 3.3; // Conversion en tension réelle (avancer-reculer)

meas\_LR = analog\_in2.read();

tension\_LR = meas\_LR \* 3.3; // Gauche-droite

printf("%lf\t",tension\_LR);

// On partitionne les valeurs de tensions en caractères

if(tensionUD>1.8){ // Avance lentement

varUD='a';

}

if(tensionUD>2.4){ // Avance normal

varUD='e';

}

if(tensionUD>3.0){ // Avance turbo

varUD='f';

}

if(tensionUD<1.4){ // Recule lentement

varUD='g';

}

if(tensionUD<0.6){ // Recule normal

varUD='b';

}

if(tensionUD >= 1.4 && tensionUD <= 1.8){ // N’avance pas

varUD='o';

}

if(tension\_LR>1.8){ // Tourne droite

varLR='r';

}

if(tension\_LR<1.4){ // Tourne gauche

varLR='l';

}

if(tension\_LR >= 1.4 && tension\_LR <= 1.8){

varLR='o';

}

tab[0]=varUD; // Variable avancer-reculer

tab[1]=varLR; // Variable gauche-droite

if(bp\_int == 1){ // Fonction envoyer signal

nRF24\_mod.setRfFrequency(2400);

nRF24\_mod.write( NRF24L01P\_PIPE\_P0, tab, 8 );

debug\_pc.printf( "SENDED\r\n");

wait\_us(100000);

}

}

}

CODE RECEPTEUR

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Test MOD-24LR / nrf24L01 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* LEnsE / Julien VILLEMEJANE / Institut d'Optique Graduate School \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Brochage \*/

/\* TO COMPLETE \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Test réalisé sur Nucléo-F411RE \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "mbed.h"

#include "MOD24\_NRF.h"

#include "nRF24.h"

PwmOut dirG1(PB\_8);

PwmOut dirG2(PC\_9);

DigitalOut en(D3);

PwmOut dirD2(PB\_14);

PwmOut dirD1(PB\_13);

Serial debug\_pc(USBTX, USBRX);

InterruptIn bp\_int(USER\_BUTTON);

/\* Main Function \*/

int main() {

initNRF24();

char meas;

char meas\_LR;

dirD1.period\_ms(10);

dirD2.period\_ms(10);

dirG1.period\_ms(10);

dirG2.period\_ms(10);

en = 1;

while(1) {

/\* Lecture donnée depuis nRF24 \*/

if ( nRF24\_mod.readable() ) {

// ...read the data into the receive buffer

rxDataCnt = nRF24\_mod.read( NRF24L01P\_PIPE\_P0, dataReceived, 8);

meas = dataReceived[0]; //Lit le signal envoyé pour avant-arrière (une lettre)

meas\_LR = dataReceived[1]; //Lit le signal envoyé pour gauche-droite (une lettre)

// Display the receive buffer contents via the host serial link

debug\_pc.printf("\tD = ");

for ( int i = 0; i < rxDataCnt; i++ ) {

debug\_pc.printf(" %x \t", dataReceived[i]);

}

debug\_pc.printf("\r\n");

if(meas == 'a'){ // Avance lentement

dirG1.write(0.3);

dirG2.write(0);

dirD1.write(0.3175);

dirD2.write(0);

}

if(meas == 'e'){ // Avance normal

dirG1.write(0.6);

dirG2.write(0);

dirD1.write(0.6175);

dirD2.write(0);

}

if(meas == 'f'){ // Avance turbo

dirG1.write(0.9);

dirG2.write(0);

dirD1.write(0.9+0.0175);

dirD2.write(0);

}

if(meas == 'b'){ // Recule normal

dirG1.write(0);

dirG2.write(1-0.5);

dirD1.write(0);

dirD2.write(1-0.5);

}

if(meas == 'g'){ // Recule lentement

dirG1.write(0);

dirG2.write(1-0.8);

dirD1.write(0);

dirD2.write(1-0.8);

}

if(meas == 'o'){ // Repos

dirG1.write(0);

dirG2.write(0);

dirD1.write(0);

dirD2.write(0);}

if(meas\_LR=='r'){ // Tourne droite

dirD1.write(0);

dirD2.write(0);

}

if(meas\_LR=='l'){ // Tourne gauche

dirG1.write(0);

dirG2.write(0);

}

}

}

}