//===================================================

//================== TaskControl =======================

void TaskControl(void \*pvParameters) // This is a task.

{

(void) pvParameters;

/\*

TaskControl: calcula la posicion en (x,y) que debe tener el circulo en su desplazamiento

\*/

byte info[4]; //dr:info[0],dd:info[1],dl:info[2],du:info[3];

//comunicada desde TaskReceive mediante mensaje

byte info1[1]; //'R':right;'L':left;'U':up;'D':down;'S':stop; comunicada a TaskTransmit mediante mensaje

portBASE\_TYPE xStatus; //estado de la cola de mensajes

enum state\_type{s0,s1,s2,s3,s4,s5,s6,s7}; //agregar mas estados de ser necesario

state\_type state;

bool Fsm\_isActive=false;

int dr,dd,dl,du; //dr:distance\_right;dd:distance\_down;dr:distance\_left;dr:distance\_up;

for (;;) // A Task shall never return or exit.

{

if (Fsm\_isActive==false){

Fsm\_isActive=true; //activar la FSM

state=s0;

}else{

//recibir mensaje enviado por TaskReceive

do{

xStatus=xQueueReceive(xQueue,info,0); //recibir el mensaje al principio de la cola, si no esta vacia

}while (xStatus != pdPASS); //mientras la cola no tenga mensaje: repetir

dr=info[0];

dd=info[1];

dl=info[2];

du=info[3];

xQueueSendToBack(xQueue1, info1, 0);

state=s0;

//control del movil:

switch(state){

//ToDo:

case s0:

xStatus=xQueueReceive(xQueue,info,0);

if (dr > 10){

info1[0] = 'R';

xQueueSendToBack(xQueue1, info1, 0);

}

if (dr <= 10){

info1[4] = 'S';

// xQueueSendToBack(xQueue1, info1, 0);

state = s1;

}

break;

case s1:

xStatus=xQueueReceive(xQueue,info,0);

if (dr > 10){

info1[3] = 'D';

xQueueSendToBack(xQueue1, info1, 0);

}

if (dr<=10){

info1[4] = 'S';

xQueueSendToBack(xQueue1, info1, 0);

state = s2;

}

break;

case s2:

xStatus=xQueueReceive(xQueue,info,0);

if (dr > 10){

info1[0] = 'R';

xQueueSendToBack(xQueue1, info1, 0);

}

if (dr<=10){

info1[4] = 'S';

xQueueSendToBack(xQueue1, info1, 0);

state = s3;

}

break;

case s3:

xStatus=xQueueReceive(xQueue,info,0);

if (dd > 10){

info1[2] = 'U';

xQueueSendToBack(xQueue1, info1, 0);

}

if (dd<=10){

info1[4] = 'S';

xQueueSendToBack(xQueue1, info1, 0);

state = s4;

}

break;

case s4:

xStatus=xQueueReceive(xQueue,info,0);

if (dd > 10){

info1[0] = 'R';

xQueueSendToBack(xQueue1, info1, 0);

}

if (dd<=10){

info1[4] = 'S';

xQueueSendToBack(xQueue1, info1, 0);

state = s5;

}

break;

case s5:

xStatus=xQueueReceive(xQueue,info,0);

if (dd > 10){

info1[3] = 'D';

xQueueSendToBack(xQueue1, info1, 0);

}

if (dd<=10){

info1[4] = 'S';

xQueueSendToBack(xQueue1, info1, 0);

state = s6;

}

break;

case s6:

xStatus=xQueueReceive(xQueue,info,0);

if (dd > 10){

info1[1] = 'L';

xQueueSendToBack(xQueue1, info1, 0);

}

if (dd<=10){

info1[4] = 'S';

xQueueSendToBack(xQueue1, info1, 0);

state = s7;

}

break;

case s7:

xStatus=xQueueReceive(xQueue,info,0);

if (dd > 10){

info1[3] = 'D';

xQueueSendToBack(xQueue1, info1, 0);

}

if (dd<=10){

info1[4] = 'S';

xQueueSendToBack(xQueue1, info1, 0);

}

}//end switch

}//end if

vTaskDelay(35 / portTICK\_PERIOD\_MS ); //esperar 100 ms en el estado bloqueado

}//end for

}//end task

//===========================================================