ERTS Final Assessment – Task 2 Presentation & Demo

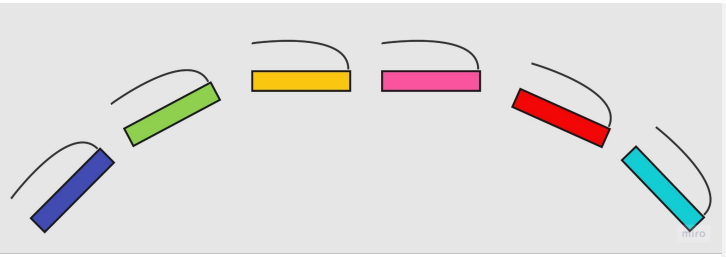
Group 15

Gytis Budrevicius	km21822
Smita Yashwant Nangare	fc23356
Hongbing Qiu	mk20661
Yumu Xie	po21744

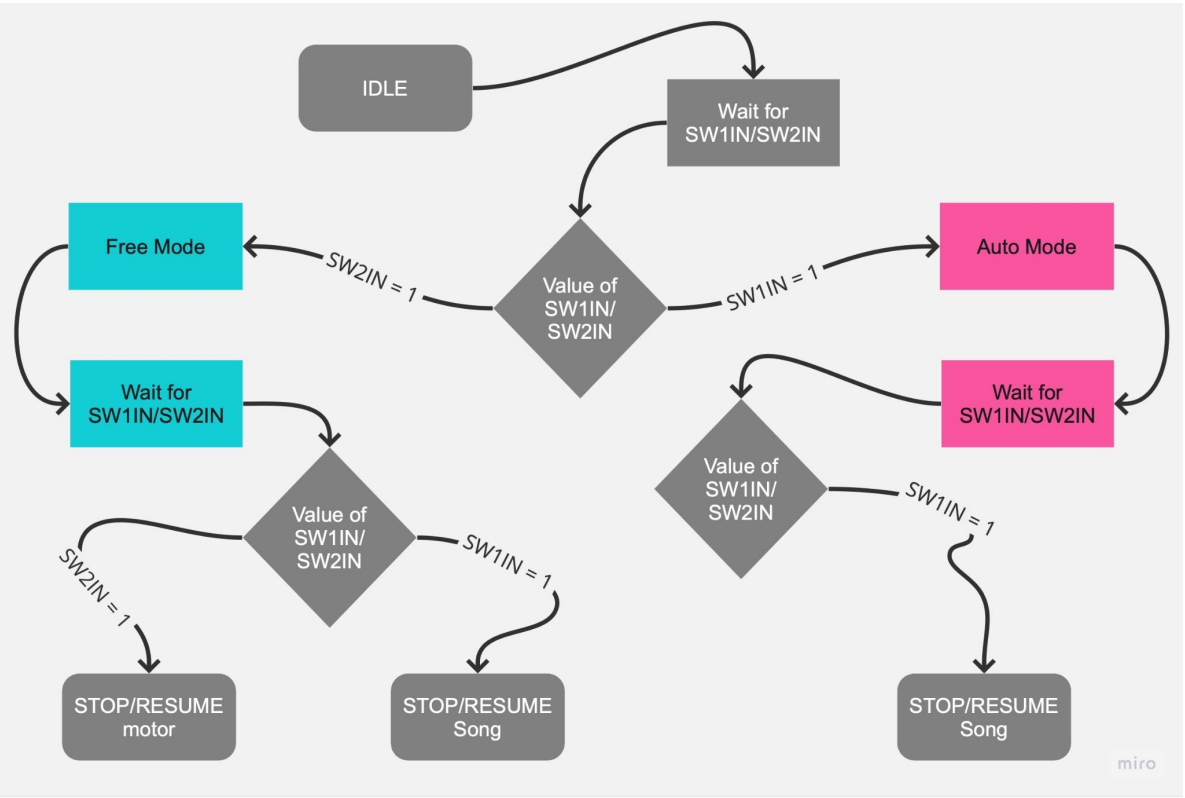
Practical elements

Show how the robot works:

Movement LED indicators:



Operation Mode Selection:



Tasks/Scheduling

What tasks have you created?

1. **Master Thread**

```
xTaskCreate(taskMasterThread, "taskT", 128, NULL, 2, &taskHandle_BlinkRedLED);
```

2. **Play song**

```
xTaskCreate(taskPlaySong,"taskS", 128, NULL, 1, &taskHandle_PlaySong);
```

3. **Bump Switches**

```
xTaskCreate(taskBumpSwitch, "taskB", 128, NULL, 1, &taskHandle_BumpSwitch );
```

4. **DC Motor**

```
xTaskCreate(taskdcMotor, "taskM", 128, NULL, 1, &taskHandle_dcMotor);
```

5. **Read Input Switches**

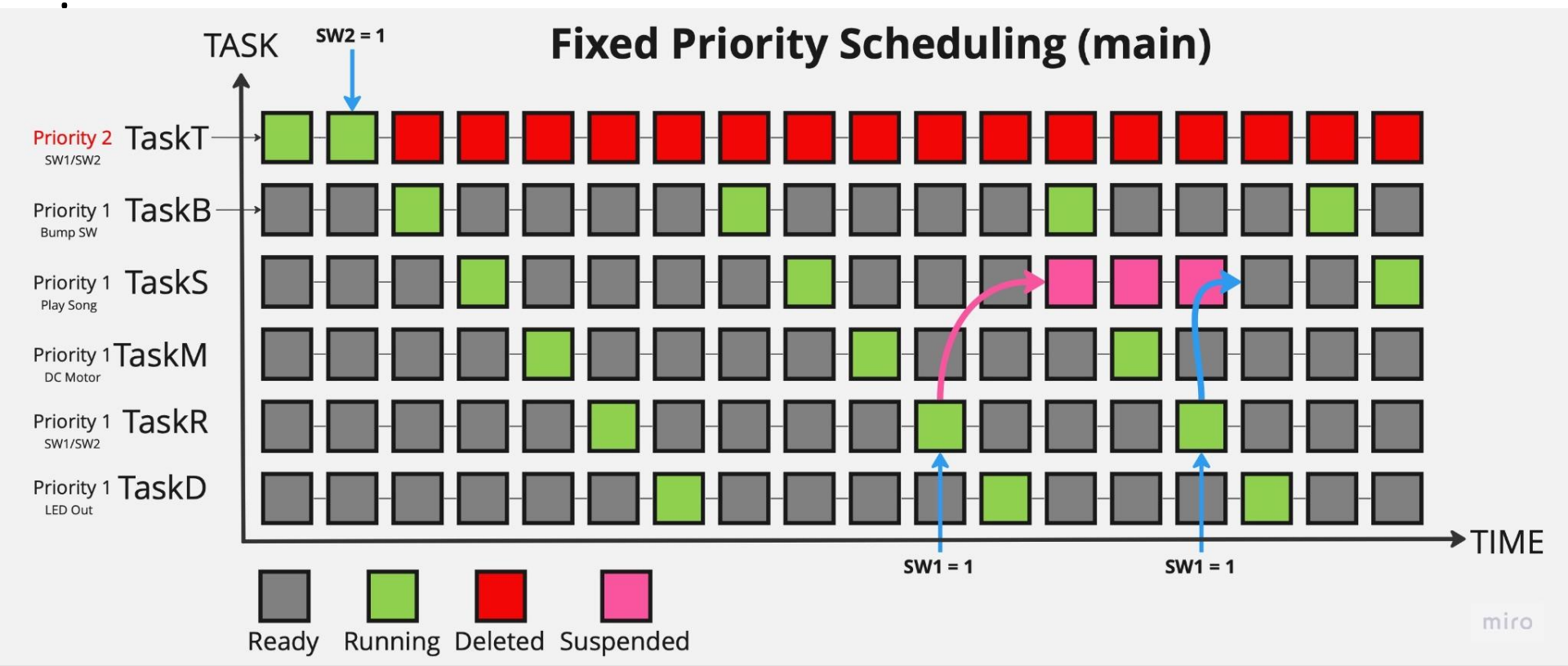
```
xTaskCreate(taskReadInputSwitch,"taskR", 128, NULL, 1, &taskHandle_InputSwitch);
```

6. **Display output LED**

```
xTaskCreate( taskDisplayOutputLED,"taskD", 128, NULL, 1, &taskHandle_OutputLED);
```

University of BRISTOL

Tasks/Scheduling



University of BRISTOL

Tasks/Scheduling

Resource control

- How have you controlled access to any shared data/hardware?

```
// TODO: declare a global variable to read bump switches value,
//       name this as bumpSwitch_status and use uint8_t
uint8_t bumpSwitch_status;
SemaphoreHandle_t xSemaphoreSwitch;
Move_mode move_mode = INIT;
uint8_t auto_stop = 0;

void main_program( void )

//initialise the switch semaphore
xSemaphoreSwitch = xSemaphoreCreateBinary();

// TODO: start the scheduler
vTaskStartScheduler();
```

threads created
schedule threads to work

```
// a static void function for taskMasterThread
static void taskMasterThread( void *pvParameters )
```

```
xSemaphoreGive(xSemaphoreSwitch);
```

semaphore give: block SW1IN & SW2IN Priority: 2
to prevent taskReadInputSwitch()
reads SW1IN & SW2IN early

```
// a static void function for taskReadInputSwitch
static void taskReadInputSwitch( void *pvParameters ){
    xSemaphoreTake(xSemaphoreSwitch,portMAX_DELAY);
```

semaphore take: release SW1IN & SW2IN Priority: 1
after take, SW1IN & SW2IN start to record
taskReadInputSwitch() starts to work

What is the difference between suspend and delete task and what are their uses ?

Suspend Task - vTaskSuspend();

Temporarily pauses a task's execution without terminating its existence, allowing it to be resumed later.

Preserves the task's resource allocation and memory, maintaining its state for eventual continuation.

Resumption is facilitated exclusively through the xTaskResume() function, providing controlled task management.

Ideally employed for tasks that require intermittent attention, allowing prioritization of other processes.

Delete Task - vTaskDelete();

Permanently removes a task from the system, effectively ending its lifecycle and operations.

Upon deletion, it relinquishes all allocated resources and memory.

The deletion process is irreversible; once a task is deleted, it cannot be revived or resumed.

Best utilized for tasks that have fulfilled their purpose and are no longer necessary, ensuring efficient task utilization.