Exercises FreeRTOS

Exercise 1:

- Open xsdk and create a new application Project.
- Provide Project name
- In OS platform select FreeRTOS10_Xilinx
- Press Next, select FreeRTOS_Hello_world. Press Finish.
- In project explorer right click on the project name - Build Project
- Open the freeRtos_hello_world file
- Identify the main function
- Identify the taskCreate functions
  - Which task has the highest priority?

- Identify the function xQueueCreate
  - How many elements can be stored in the queue?

  Connect the serial port and Run the code
- Right click again on the project name
- Go to Run As – Launch on Hardware (GDB)

Exercises FreeRTOS

Exercise 2:

- Invert the priorities of the task
  - What should happen?

    Run the code and verify

- Add a vTaskDelay(x1second) to the RxTask to force it to go to the Blocked State
  - What should happen now?

    Run the code and verify

- Put the same prority to both tasks and run the code.

Exercises FreeRTOS

Exercise 3:

- Modify the number of items to be stored in the queue to ten.
- Use the following function in the RX task to see how many messages are waiting in the queue

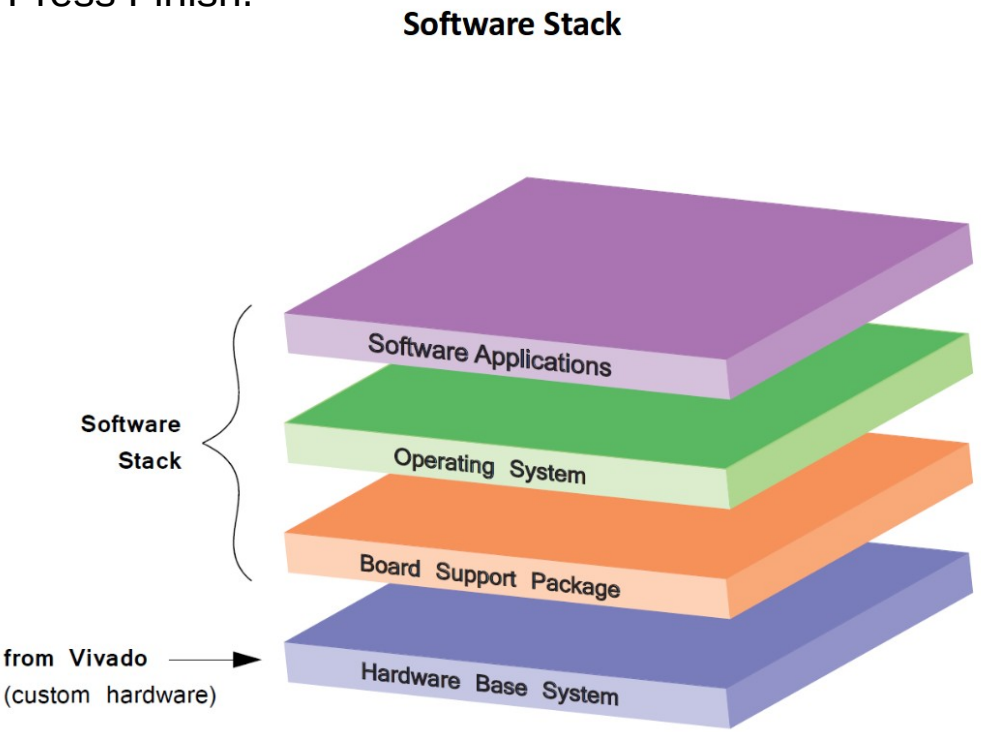
```
tUBaseType_t uxQueueMessagesWaiting( QueueHandle_t xQueue );
```

Run the code and analyze the obtained response

Exercises FreeRTOS

Exercise 4:

- Create a project in Vivado with a Zynq processor and a GPIO to the leds
- Generate Bitstream and export hardware including bitstream
- Open xsdk and create a new application Project.
- Provide Project name
- In OS platform select FreeRTOS_10_Xilinx
- Press Next, select FreeRTOS_Hello_world. Press Finish.



Exercises FreeRTOS

Exercise 4 (cont):

- Change the main function for

```
int main ( void ){
    // do needed Platform initialization
    // 1) Start LED 1 toogle
    xTaskCreate(Task_LED,  (signed char*) "LEDs", 1024, NULL, 1, NULL);

   //  2) printf
    xTaskCreate (Task_Print, (signed char*) "Print", 1024, NULL, 1, NULL);

   // Finally: Start FreeRTOS
    vTaskStartScheduler();

// Will only reach here if there was insufficient memory to create the idle task
    return 0;
}
```

Exercises FreeRTOS

Exercise 4 (cont):

- Add the corresponding tasks

```
int tick=0;
```

```
void Task_LED (void* p)
{
int tick;
  while (1)
  {
     Xil_Out32 (aGPIO, tick;
     vTaskDelay (100);
  }
}
```

```
void Task_Print (void* p)
{
  while (1)
  {
     printf („Tick is %d \n“, tick);
     vTaskDelay (500);
     tick++;

  }
}
```

Default GPIO Base-Address: look in xParameters.h (0x4120_0000)

Configure the FPGA and Run the code