

FWD – Advanced Embedded Systems Course

Project #2 – RTOS

❖ Rubric Requirement4 : “Verify the system Implementation”

➤ System Hyperperiod Calculation :

- Task1 Periodicity = 50
 - Task2 Periodicity = 50
 - Task3 Periodicity = 100
 - Task4 Periodicity = 20
 - Task5 Periodicity = 10
 - Task6 Periodicity = 100
 - ⇒ Hyperperiod = 100ms , The least significant multiplier of all tasks periodicity
- System CPU Load :

Tasks practical Periodicity and Execution time:

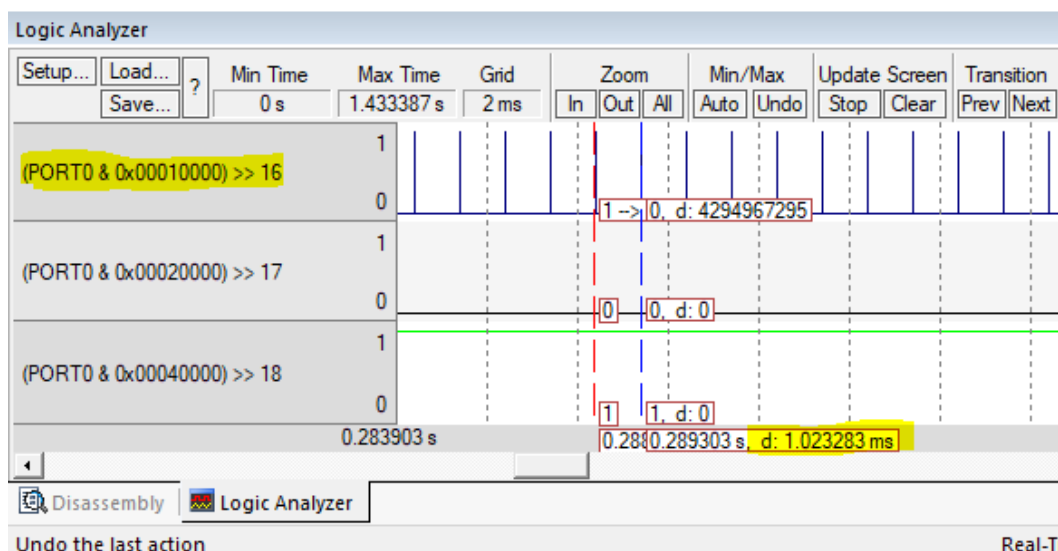
- Tick:
- > Tick period = 1ms

The screenshot shows the IDE interface. On the left, the 'Registers' window displays a list of registers with their values. The 'main.c' file is open in the center, showing the `vApplicationTickHook` function. The function calls `GPIO_write` to set the pin to high and then low. Below the code editor, the 'Watch 1' window shows the value of `EventGroupVar` as `0x00000000`.

Register	Value
R 0x00000000	0x00000000
R 0x01010101	0x01010101
R 0x02020202	0x02020202
R 0x0000154B	0x0000154B
R 0x04040404	0x04040404
R 0x05050505	0x05050505

```
162 }
163 }
164 /*-----
165
166 void vApplicationTickHook( void )
167 {
168     GPIO_write(PORT_0,PIN0, PIN_IS_HIGH);
169     GPIO_write(PORT_0,PIN0, PIN_IS_LOW);
170 }
```

Name	Value	Type
EventGroupVar	0x00000000	uint
<Enter expression>		



- Task 1
→ Periodicity = 50 ms

The screenshot shows the IDE with the following components:

- Registers:** A list of registers with values. The current register (Cu...) is 0x00000000.
- main.c:**

```

154 xQueueSend( xQueue1, ( void * ) &EventGroupVar, ( TickType_t ) 10 );
155 /*Update Variables*/
156 B1_Old_State = B1_Current_State ;
157
158 GPIO_write(PORT_0,PIN1, PIN_IS_LOW);
159 /*Block Task 50ticks (50ms)*/
160 vTaskDelayUntil( &xLastWakeTime, (TickType_t)50 );
161 GPIO_write(PORT_0,PIN1, PIN_IS_HIGH);
162

```
- Watch 1:**

Name	Value	Type
EventGroupVar	0x00000000	uint
<Enter expression>		
- Logic Analyzer:**
 - Signals: (PORT0 & 0x00010000) >> 16, (PORT0 & 0x00020000) >> 17, (PORT0 & 0x00040000) >> 18.
 - Time scale: 10 ms.
 - Time markers: 0.246563 s, 0.250301 s, 0.300363 s, 0.356563 s.
 - Highlighted delay: 50.06232 ms.

Real-Time Agent: Target Stopped

→ Execution Time = 13.9 us

The screenshot shows the IDE with the following components:

- Registers:** A list of registers with values. The current register (Cu...) is 0x00000000.
- main.c:**

```

154 xQueueSend( xQueue1, ( void * ) &EventGroupVar, ( TickType_t ) 10 );
155 /*Update Variables*/
156 B1_Old_State = B1_Current_State ;
157
158 GPIO_write(PORT_0,PIN1, PIN_IS_LOW);
159 /*Block Task 50ticks (50ms)*/
160 vTaskDelayUntil( &xLastWakeTime, (TickType_t)50 );
161 GPIO_write(PORT_0,PIN1, PIN_IS_HIGH);
162

```
- Watch 1:**

Name	Value	Type
EventGroupVar	0x00000000	uint
<Enter expression>		
- Logic Analyzer:**
 - Signals: (PORT0 & 0x00010000) >> 16, (PORT0 & 0x00020000) >> 17, (PORT0 & 0x00040000) >> 18.
 - Time scale: 2 us.
 - Time markers: 0.300284 s, 0.300288 s, 0.300302 s.
 - Highlighted delay: 13.88333 us.

Real-Time Agent: Target Stopped

- Task2 : Same as Task1
→ Periodicity = 50 ms
→ Execution Time =13.9 us

- Task3 :
→ Periodicity = 100ms (0.1s)

The screenshot displays a development environment with three main components:

Registers

Register	Value
R0	0x00000001
R1	0x01010101
R2	0x02020202
R3	0x000014B7
R4	0x04040404
R5	0x05050505
R6	0x06060606
R7	0x07070707
R8	0x08080808
R9	0x09090909
R10	0x10101010
R11	0x11111111
R12	0x12121212
R...	0x400007D8
R...	0x000014B7

Source Code (main.c)

```

122
123 /* task3 "Periodic Transmitter" ISR, send periodic string e
124 void Task_3( void * pvParameters )
125 {
126     for (;;) {
127         // Create the xLastWakeTime variable and Initialise it wi
128         TickType_t xLastWakeTime;
129         xLastWakeTime = xTaskGetTickCount();
130
131         //send the PeriodicString to the Queue
132         xQueueSend( xQueue1, ( void * ) &PeriodicString, ( TickType
133
134         GPIO_write(PORT_0,PIN1, PIN_IS_LOW);
135         /*Block Task 100ticks (100ms)*/
136         vTaskDelayUntil( &xLastWakeTime, (TickType_t)100 );
137         GPIO_write(PORT_0,PIN1, PIN_IS_HIGH);
138     }

```

Logic Analyzer

Setup: Min Time 0 s, Max Time 1.959351 s, Grid 50 ms. Zoom: In, Out, All. Min/Max: Auto, Undo. Update Screen: Stop, Clear. Transition: Prev, Next. Jump to: Code, Trace. Signal Info: [] Show Cycl.

Signal	Value	Time
(PORT0 & 0x00010000) >> 16	0	0.414576 s
(PORT0 & 0x00020000) >> 17	1 → 0	0.500299 s
(PORT0 & 0x00040000) >> 18	1	0.600576 s, d: 0.100277 s

Real-Time Agent: Target Stopped

→ Execution Time = 7.5 us

The screenshot displays a development environment with three main panels: Registers, Code, and Logic Analyzer.

Registers Panel: Shows the current state of registers R0 through R12. R0 contains 0x00000001, R1 contains 0x01010101, R2 contains 0x02020202, and so on. R12 contains 0x12121212.

Code Panel: Displays the source code for `main.c`. The code defines a task `Task_3` that writes to `PORT_0` and delays for 100 ticks. The execution time is noted as 7.5 us.

```
122
123 /* task3 "Periodic_Transmitter" ISR, send periodic string every 100 ms to cons
124 void Task_3( void * pvParameters )
125 {
126     for (;;) {
127         // Create the xLastWakeTime variable and Initialise it with the current time
128         TickType_t xLastWakeTime;
129         xLastWakeTime = xTaskGetTickCount();
130
131         //send the PeriodicString to the Queue
132         xQueueSend( xQueue1, ( void * ) &PeriodicString, ( TickType_t
133
134         GPIO_write(PORT_0,PIN1, PIN_IS_LOW);
135         /*Block Task 100ticks (100ms)*/
136         vTaskDelayUntil( &xLastWakeTime, (TickType_t)100 );
137         GPIO_write(PORT_0,PIN1, PIN_IS_HIGH);
138     }
129
```

Logic Analyzer Panel: Shows a timing diagram for three signals: `(PORT0 & 0x00010000) >> 16`, `(PORT0 & 0x00020000) >> 17`, and `(PORT0 & 0x00040000) >> 18`. The signals are shown as digital waveforms. The time scale is 1 us. The execution time is 0.600291 s.

General Purpose Input/Output (GPIO) Panel: Shows the configuration for GPIO0. The pins are configured as follows:

- IO0DIR: 0xEFFB0000
- IO0SET: 0x00000000
- IO0CLR: 0x00000000
- IO0PIN: 0x0004FCFF
- Pins: 0x7804FEFF

Real-Time Agent: Target Stopped

- Task4 :

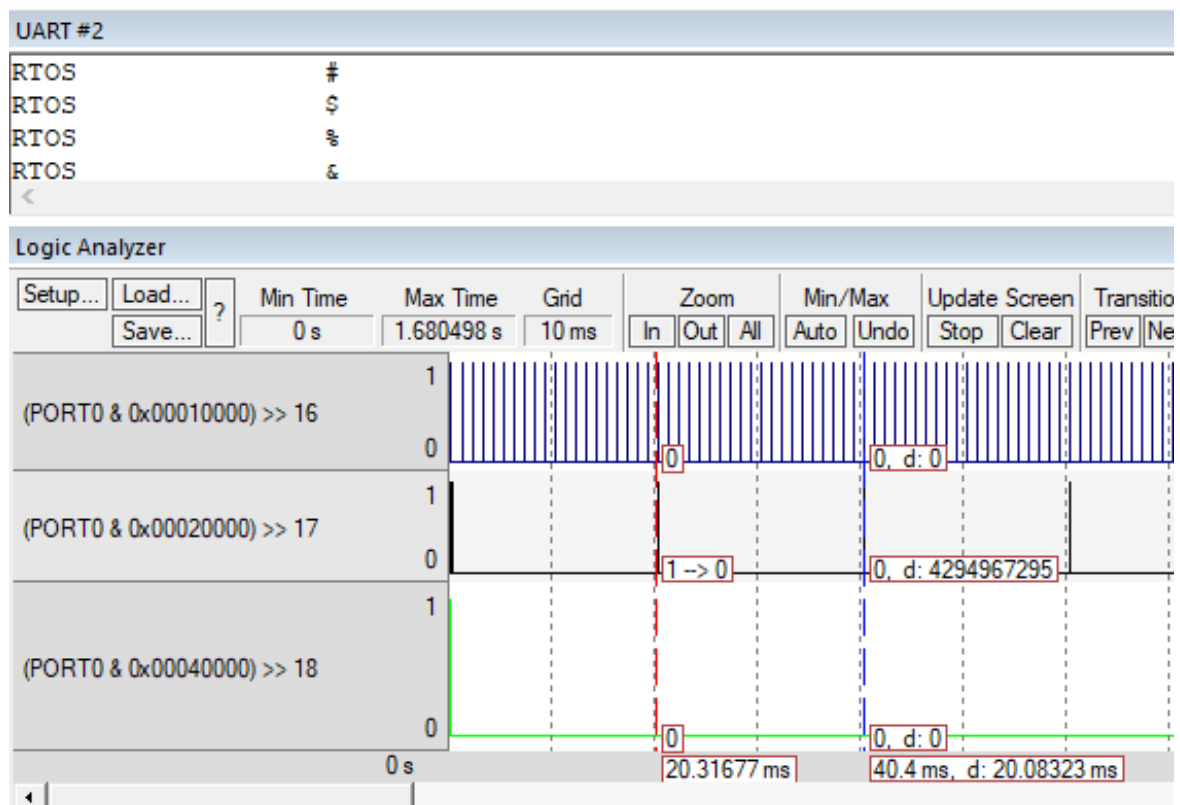
→ As we need the Queue to be full in order to test the Task actual load, i will Load the Queue every time inside the task body first, then start monitoring task with GPIOs starting from its particular job code only

```

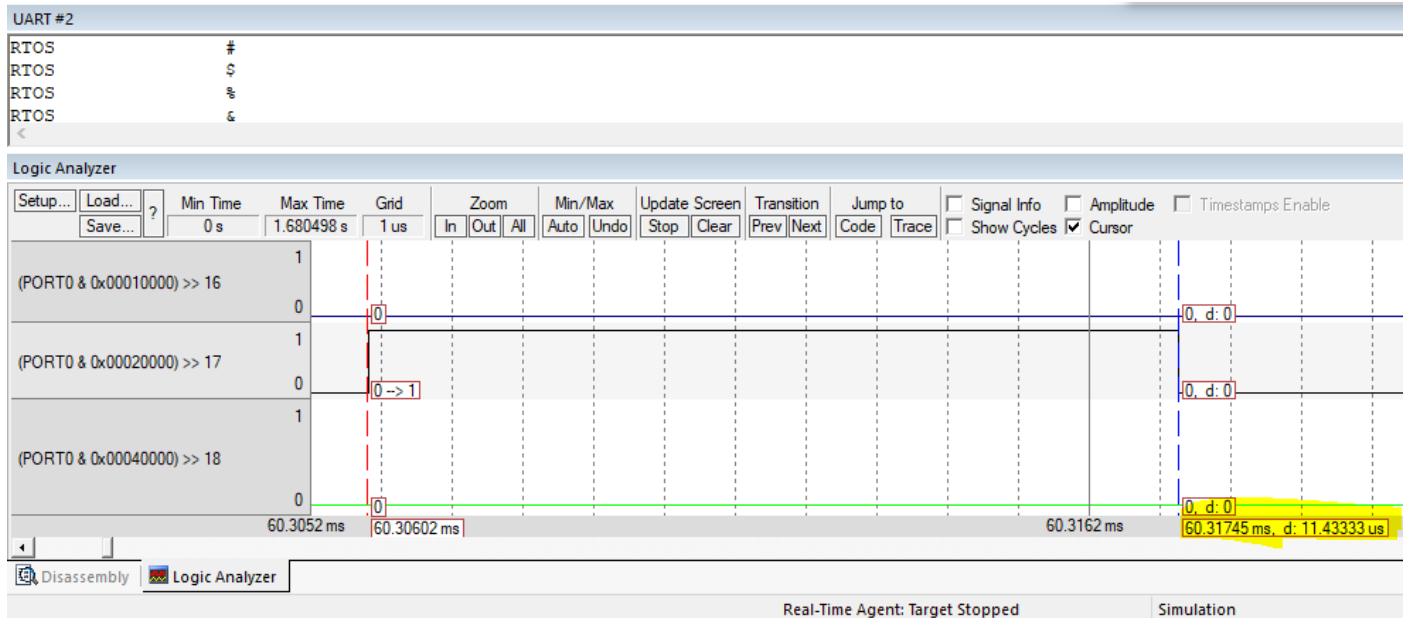
123
124 /* Task 4 "Uart_Receiver", write on uart received string from other tasks. */
125 void Task_4( void * pvParameters )
126 {
127     for (;;) {
128         // Create the xLastWakeTime variable and Initialise it with the current time for using delayuntill fn
129         TickType_t xLastWakeTime;
130         xLastWakeTime = xTaskGetTickCount();
131         // Load the Queue, Temporary for Measuring Task Load
132         xQueueSend( xQueue1, ( void * ) &PeriodicString, ( TickType_t ) 10 ); // PeriodicString = 4Bytes, xQueue1 = 20Byte
133         xQueueSend( xQueue1, ( void * ) &PeriodicString, ( TickType_t ) 10 );
134         xQueueSend( xQueue1, ( void * ) &PeriodicString, ( TickType_t ) 10 );
135         xQueueSend( xQueue1, ( void * ) &PeriodicString, ( TickType_t ) 10 ); // xQueue1 Loaded with 16Bytes
136
137         GPIO_write(PORT_0,PIN1, PIN_IS_HIGH); // Start Calculating task actual execution time
138         if( xQueue1 != NULL ) // if the Queue contains data
139         {
140             xSerialPutChar('\n');
141             xQueueReceive( xQueue1, &(ReceivedFromQueue), ( TickType_t ) 10 );
142             ReceivedFromQueue[19] += 0x01;
143             vSerialPutString(ReceivedFromQueue,20);
144         }
145         GPIO_write(PORT_0,PIN1, PIN_IS_LOW);
146         /*Block Task 20ticks (20ms)*/
147         vTaskDelayUntil( &xLastWakeTime, (TickType_t)20 );
148     }
149 }
150
151 /*-----*/

```

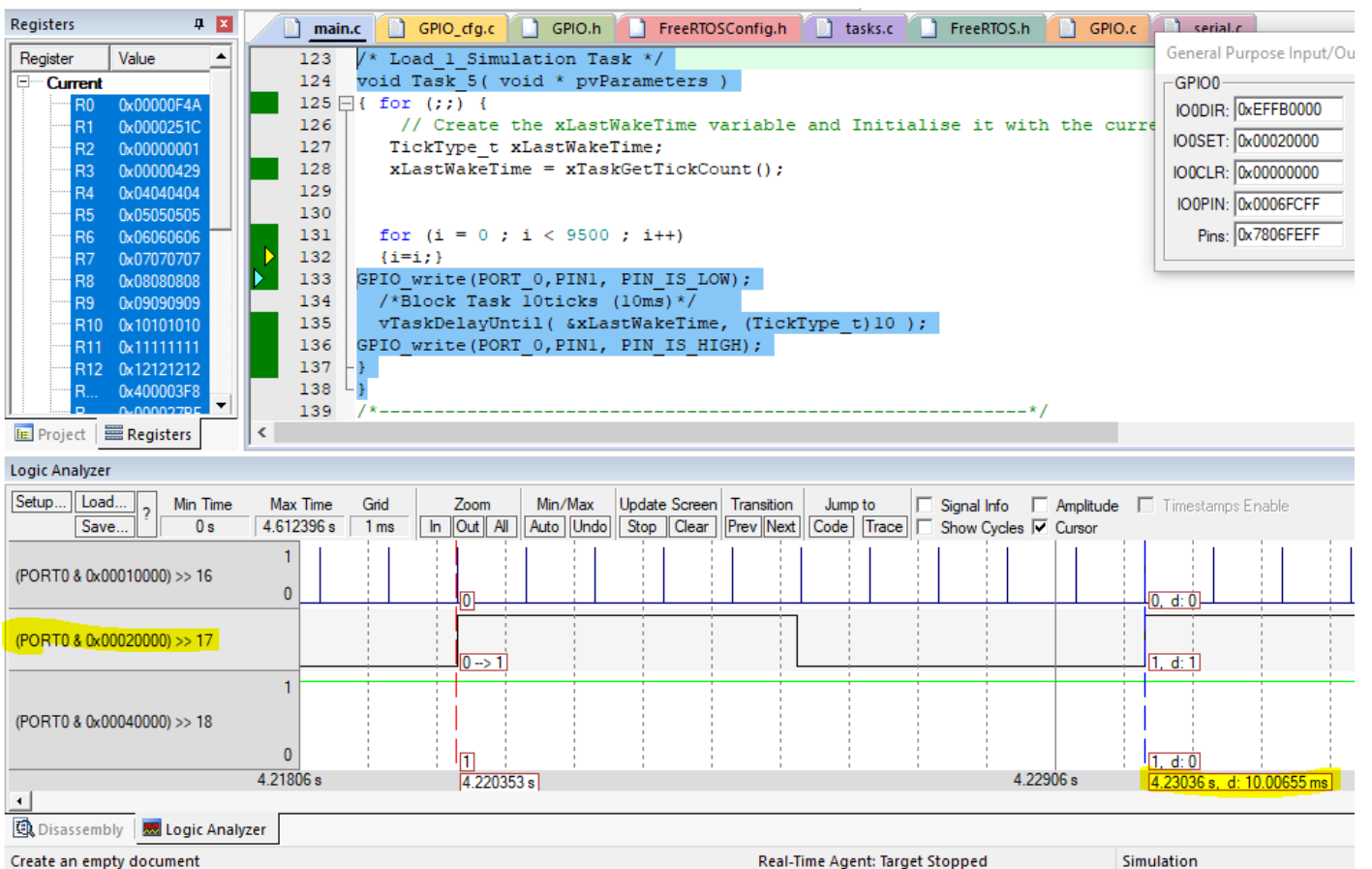
→ Periodicity = 20ms



→ Periodic time = ~ 11.5us



- Task5 :
→ Periodicity = 10ms



→ Execution ≈ 5ms

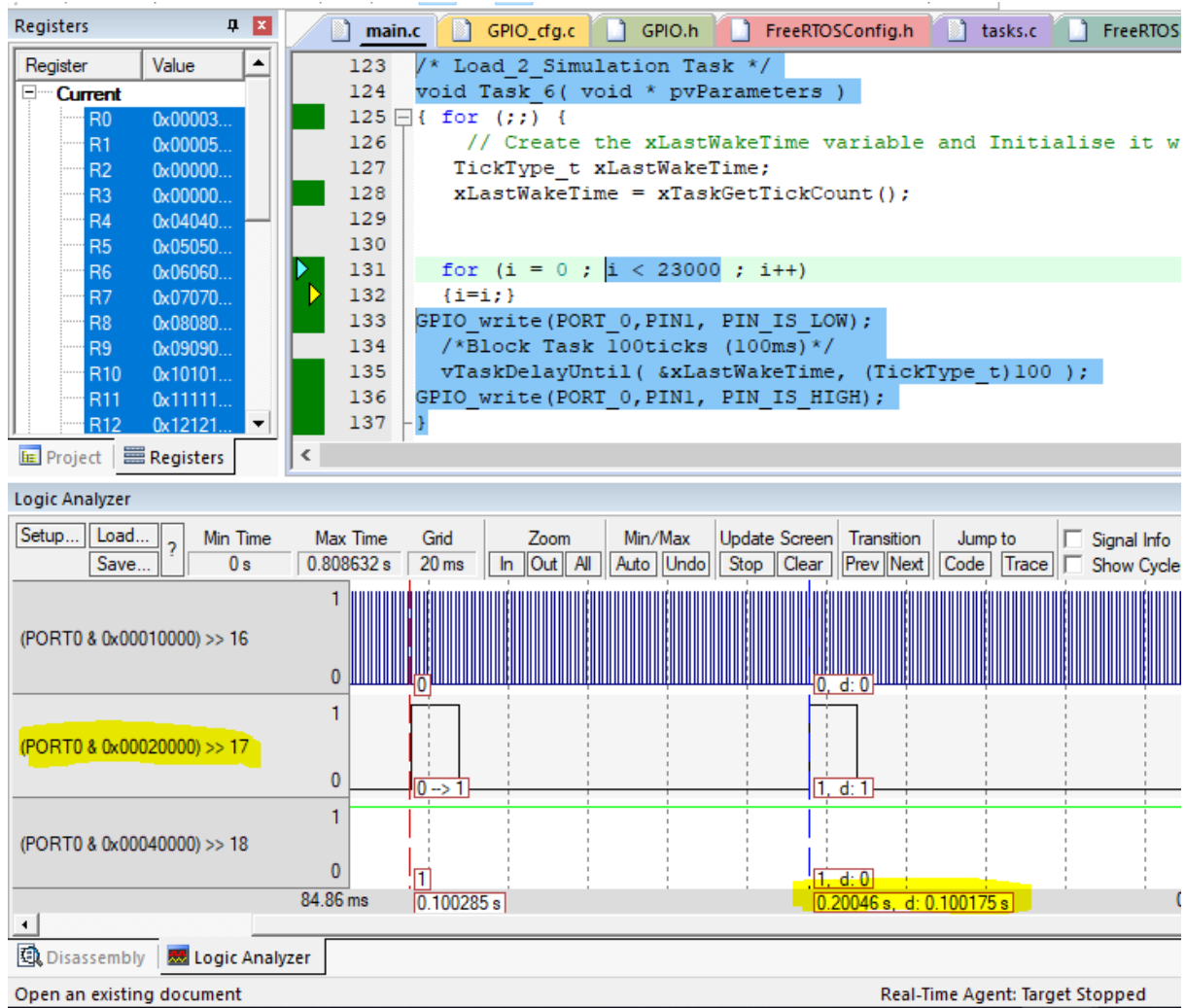
The screenshot displays a software development environment with three main panels:

- Registers Panel:** Shows the current state of registers R0 through R12. R0 is 0x00000F4A, R1 is 0x0000251C, R2 is 0x00000001, R3 is 0x00000429, R4 is 0x04040404, R5 is 0x05050505, R6 is 0x06060606, R7 is 0x07070707, R8 is 0x08080808, R9 is 0x09090909, R10 is 0x10101010, R11 is 0x11111111, R12 is 0x12121212, and R... is 0x400003F8.
- Source Code Panel:** Displays the C code for `Task_5` in `main.c`. The code includes a loop that writes to `PORT_0, PIN1` and delays for 10 ticks (10ms) using `vTaskDelayUntil`. The code is as follows:

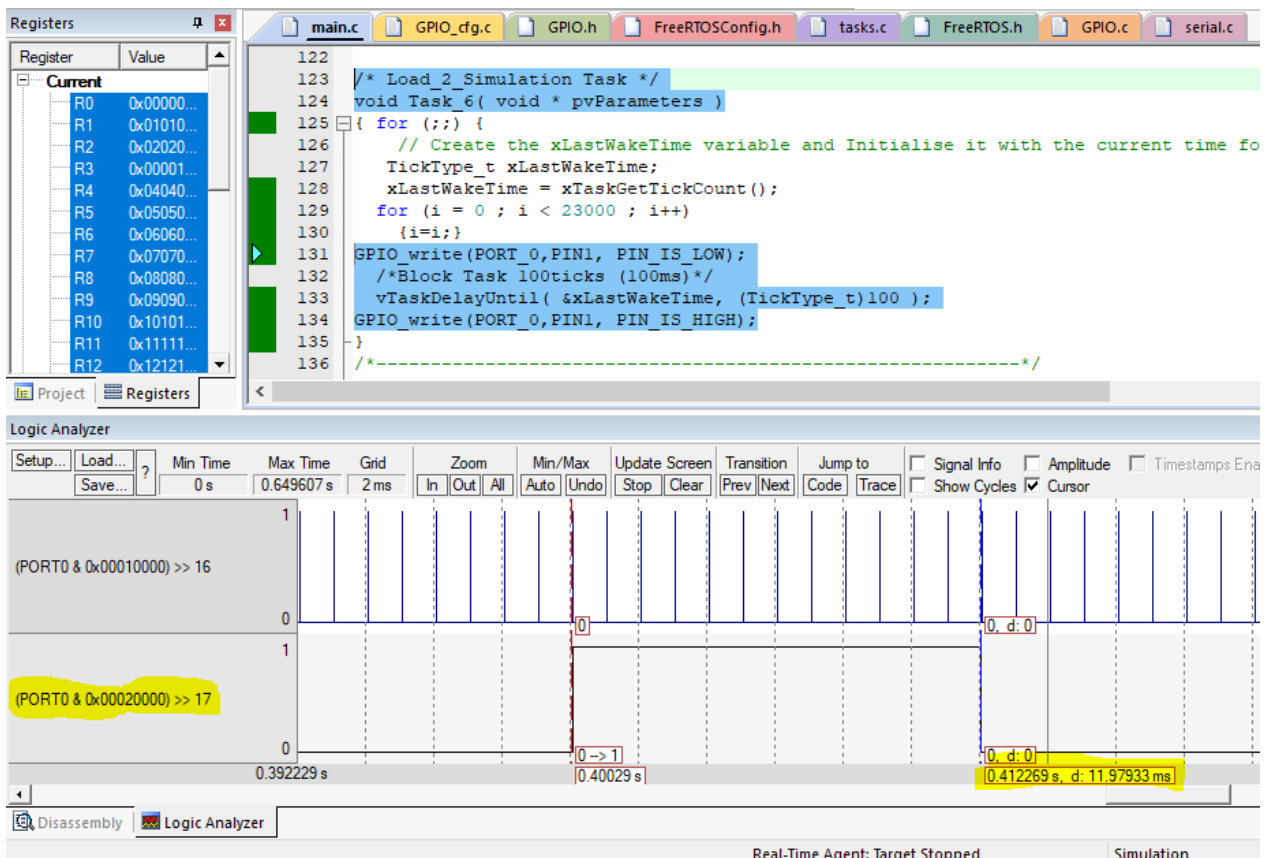
```
123 /* Load_1_Simulation Task */
124 void Task_5( void * pvParameters )
125 { for (;;) {
126     // Create the xLastWakeTime variable and Initialise it with the current
127     TickType_t xLastWakeTime;
128     xLastWakeTime = xTaskGetTickCount();
129
130
131     for ( i = 0 ; i < 9500 ; i++)
132     {i=i;}
133     GPIO_write(PORT_0,PIN1, PIN_IS_LOW);
134     /*Block Task 10ticks (10ms)*/
135     vTaskDelayUntil( &xLastWakeTime, (TickType_t)10 );
136     GPIO_write(PORT_0,PIN1, PIN_IS_HIGH);
137 }
138 }
139 /*-----*/
```
- Logic Analyzer Panel:** Shows a timing diagram for three signals: `(PORT0 & 0x00010000) >> 16`, `(PORT0 & 0x00020000) >> 17`, and `(PORT0 & 0x00040000) >> 18`. The signals are shown as digital waveforms. The time scale is 1 ms. The signals show transitions at approximately 4.21806 s, 4.22035 s, and 4.2253 s.

The bottom status bar indicates "Real-Time Agent: Target Stopped" and "Simu".

- Task6 :
→ Periodicity = 100ms



→ Execution ≈ 12ms



CPU Load Calculation :

- CPU Load = total execution time of all tasks in one hyperperiod(cpu busy time) / hyperperiod
= ((2*(13.9/1000)) + (2*(13.9/1000)) + (1* (7.5/1000)) + (5* (11.5/1000)) + (10 * (5)) + (1* (12))) / 100 =
0.621206
= 62.12 %