**Tópicos Avançados I – EC 020**

**Parte 2 – Implementação FreeRTOS**

**Fernanda Carneiro Avelar**

**João Pedro Oneti Carvalho**

**Karla Pereira do Carmo**

**Lucas de Paiva Rosa Gaspar**

Prof.º Evandro Luis Brandão Gomes

**I. OBJETIVO DO PROJETO**

O objetivo deste projeto é utilizar os fundamentos aprendidos na aula EC020 juntamente com outros conhecimentos adquiridos durante o curso de Engenharia da Computação em um projeto de teor prático envolvendo a LPCxpresso Base Board aprimorando um sistema baseado em super-loop com *Bare Metal* que envolve dois sensores, permitindo o usuário alterar funcionalidades do sistema, no qual os dados são disponibilizados em uma interface ethernet para um sistema utilizando RTOS Real Time Operating Systems.

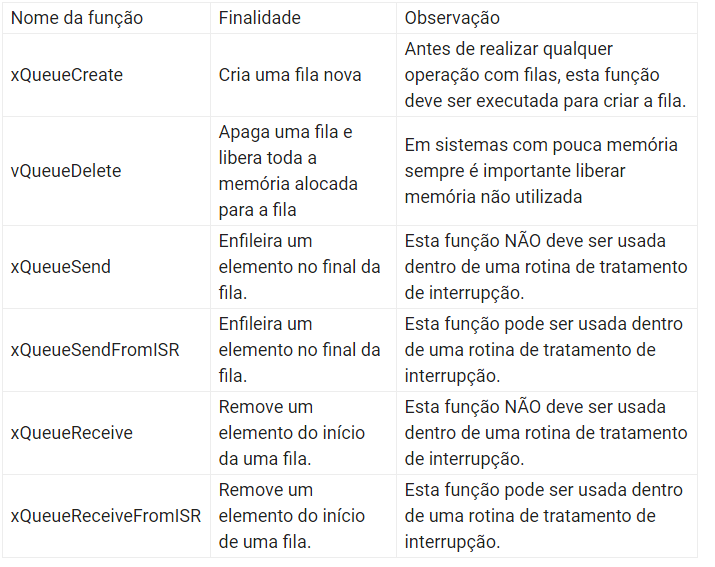
**II. DESCRIÇÃO GERAL DO PROJETO E SUAS FUNÇÕES**

O projeto é um sistema de controle dinâmico de temperatura e luminosidade de um cômodo, no qual os valores medidos pelos sensores podem ser monitorados pela interface web desenvolvida pelos integrantes do grupo. No display da placa pode ser escolhido qual medida de sensor será exibida, trocando a exibição através do botão. Para alterar os parâmetros máximos de luminosidade e de temperatura é utilizado o protocolo CC:Payload, onde o comando 01 altera a temperatura máxima e 02 altera a luminosidade máxima. Na UART é apenas exibida os dados, já na interface web, pode ser alterado a porcentagem que irá aparecer na *progress bar* do sensor.

Para a segunda parte do projeto, foi proposto a melhoria do projeto utilizando de um RTOS Real Time Operating Systems, para atender as funcionalidades do sistema e, foi implementado uma UI utilizando interface ethernet para disponibilização dos dados em uma página HTML.

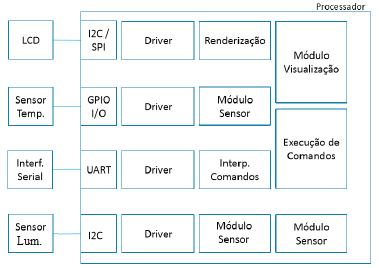
Alteramos as funções do projeto para executarem como tarefas, que são atendidas pelo sistema através de FILA, uma fila é uma estrutura em que os dados são organizados de forma sequencial. Uma fila tem um comportamento *First-In-First-Out*(FIFO) ou seja, o primeiro byte a ser inserido na fila será o primeiro a ser removido. As filas circulares são implementadas no arquivo queue.c do código fonte do FreeRTOS.

A tabela abaixo mostra as principais funções do FreeRTOS relacionados a manipulação de filas:



*Figura 1 – Funções do FreeRTOS para manipulação de FILAS*

**III. DIAGRAMA DE BLOCOS DO HARDWARE**

****

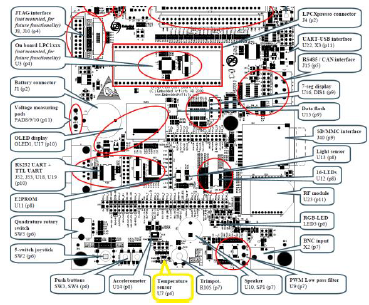
*Figura 2 – Diagrama em Blocos*

**IV. DESCRIÇÃO COMPLETA DOS SENSORES UTILIZADOS INDICANDO A INTERFACE DE COMUNICAÇÃO COM O MICROCONTROLADOR**

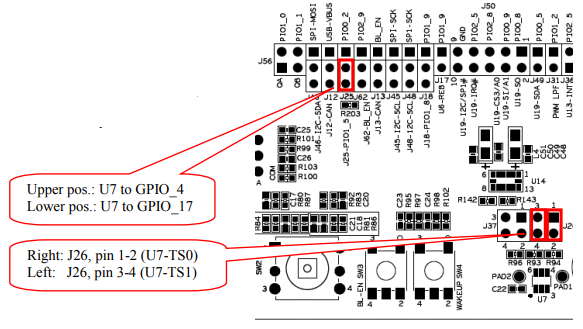
**IV.II. SENSOR DE TEMPERATURA**

O MAX6576/MAX6577 é um sensor de temperatura integrado a placa LPCXpresso e conectado ao LPC1769 através de um pino GPIO digital, podendo ser o pino PIO0\_2 ou PIO1\_5, dependendo da seleção feita pelo jumper J25. Sua faixa de temperatura varia de -40ºC à +125ºC e opera entre 2,7V e 5,5V.

A Figura 2 mostra a localização dos principais componentes do LPCXpresso e a localização do sensor de temperatura na placa marcada em amarelo.



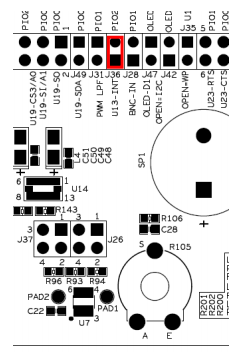
*Figura 3 - Localização do sensor de temperatura na placa*

**

*Figura 4 - Localização dos jumpers para o funcionamento do sensor de temperatura*

**IV.III. DESCRIÇÃO DO SENSOR DE LUMINOSIDADE**

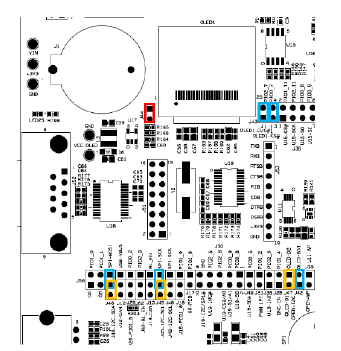
O dU13 está sempre conectado ao bus I2C. É uma saída de interrupção que pode ser conectada GPIO\_33 via J36 como indicado na *Figura 2*. É preciso certificar-se de que não há outro driver no sinal GPIO\_33 ao usar como sinal de interrupção. Existe uma série de 270 ohm resistência para evitar danos no caso de haver mais de um driver do sinal.



*Figura 5 - Localização dos jumpers para o funcionamento do sensor de luminosidade*

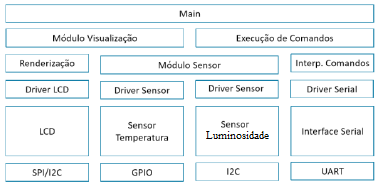
**IV.IV. DESCRIÇÃO DO DISPAY OLED**

O *display* OLED pode ser conectado utilizando o barramento SPI ou I2C, sempre inserindo um jumper em J44 para permitir que o GPIO\_29 controle a voltagem do OLED. Neste projeto foi utilizada a interface I2C e, sendo assim, os jumpers devem ser inseridos em J45, J46 e J47, pinos 2-3, como ilustra a Figura 3 nas marcações em amarelo.

**

*Figura 6 - Localização dos jumpers para o funcionamento do OLED*

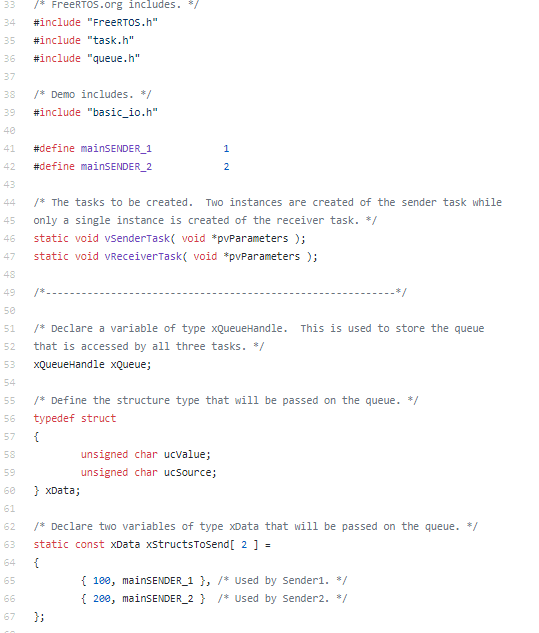
**V. DIAGRAMA DE ARQUITETURA DE SOFTWARE (CAMADAS)**



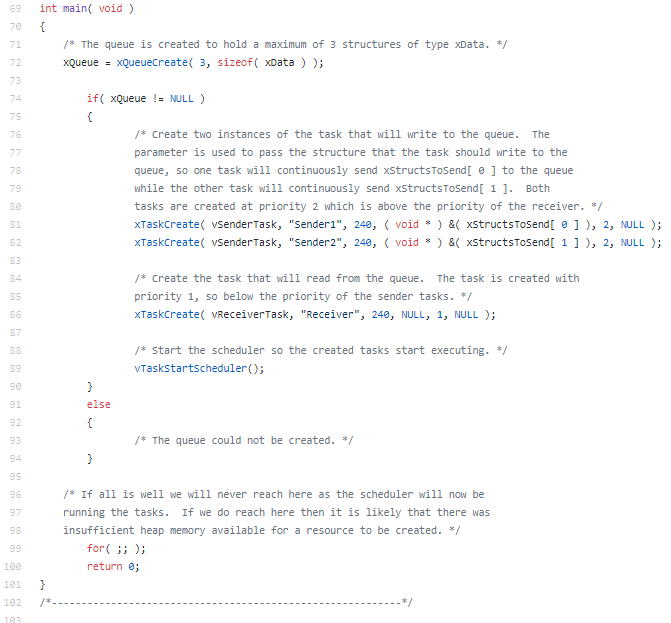
*Figura 7 – Diagrama em Camadas*

**VI. O SOFTWARE IMPLEMENTADO**

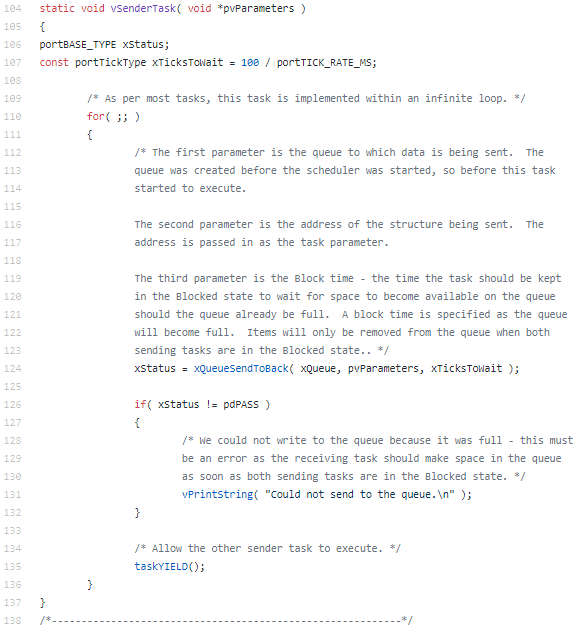
**VI.I INTEGRAÇÃO DO FREERTOS:**

****

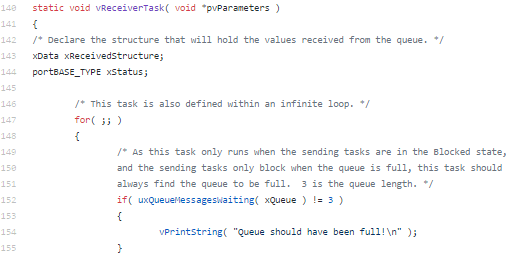
*Figura 8 – Biblioteca e Variáveis para RTOS*

**­­**

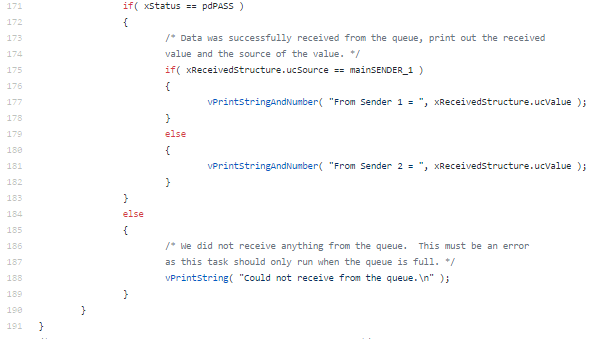
*Figura 9 – Implementação de Fila RTOS*

****

*Figura 10 – Função VSenderTask*

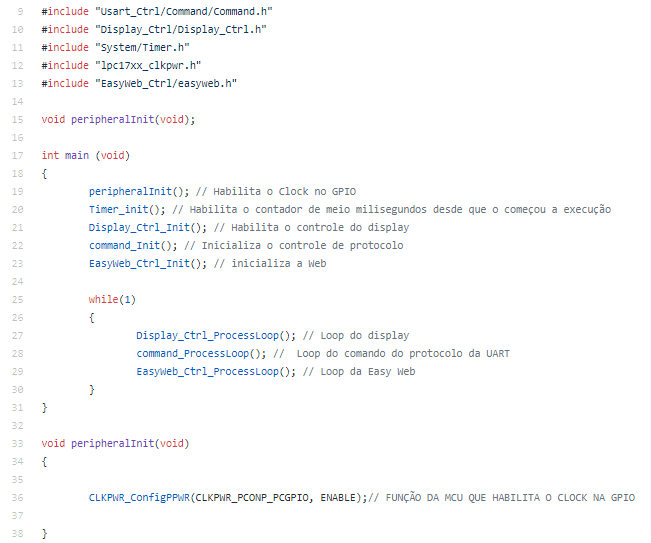
****

*Figura 11 – Função vReceiverTask*

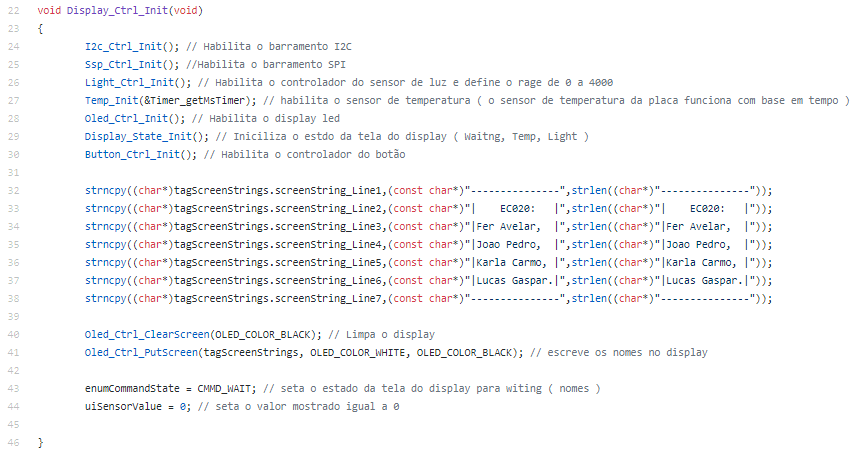
****

*Figura 12 – Função vReceiverTask*

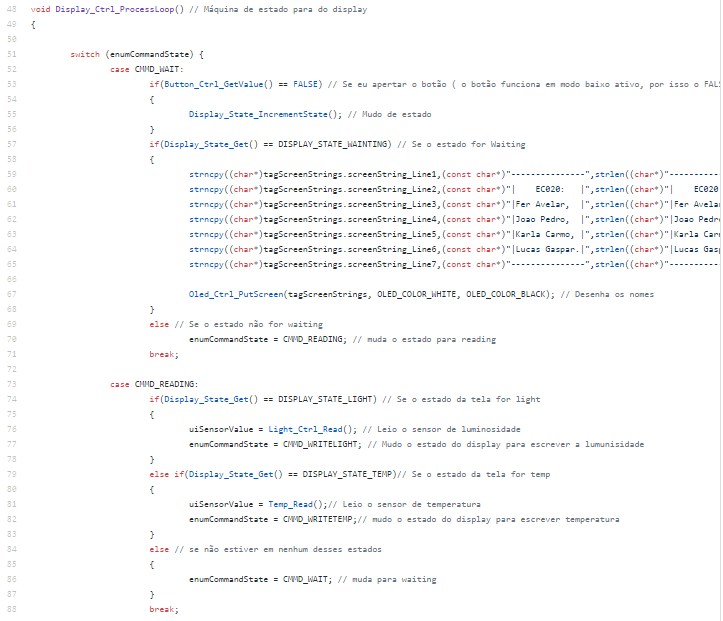
**IMPLEMENTAÇÃO COM BARE METAL:**

****

*Figura 13 – Código da função principal*

**

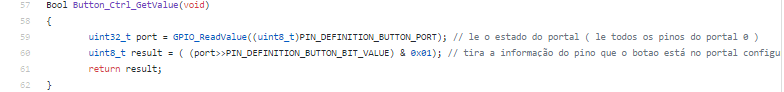
*Figura 14 – Código da função Display\_Ctrl\_Init*

**

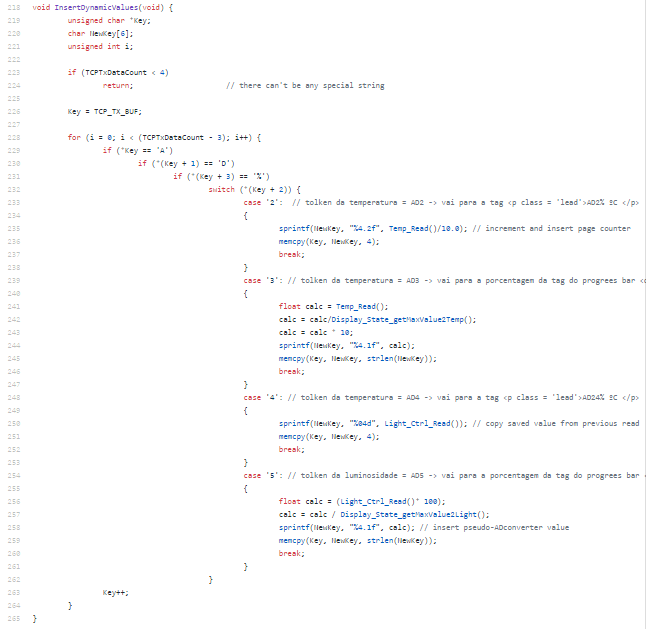
*Figura 15 – Código da função Display\_Ctrl\_ProcessLoop responsável por controlar a máquina de estado do display*

**

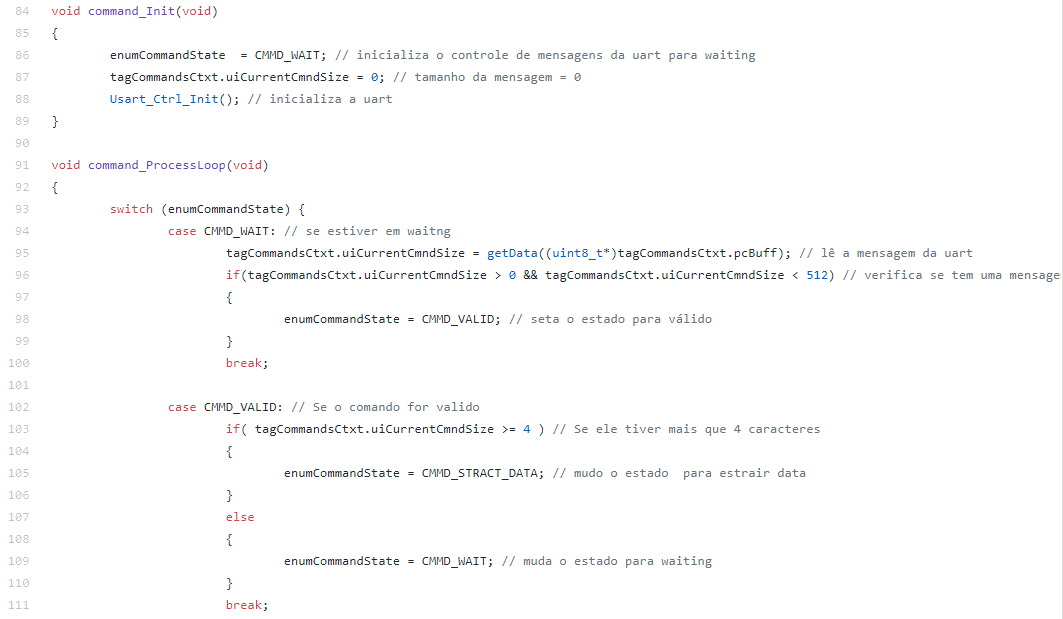
*Figura 16 – Estados da função Display\_Ctrl\_ProcessLoop*

**

*Figura 17 – Código da função Button\_Ctrl\_GetValue*

****

*Figura 18 – Código da função InsertDynamicValues*

****

*Figura 19 – Código da função command\_Init e Command\_ProcessLoop*

**

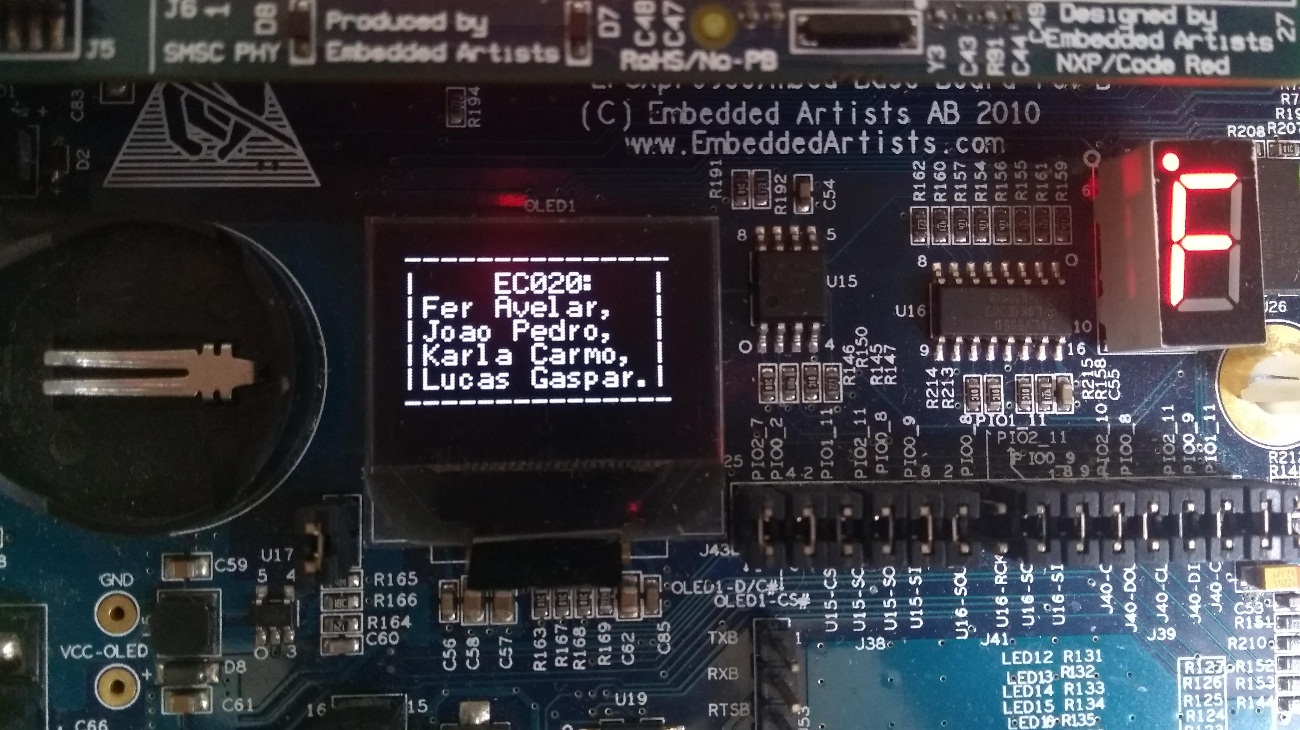
*Figura 20 – Alguns estados da função Command\_ProcessLoop*

**VII. RESULTADOS OBTIDOS**

Com base nos dados lidos, o sensor detecta os valores de temperatura e de luminosidade do cômodo e imprime este valor no *display*. Os dados também são exibidos na interface ethernet.

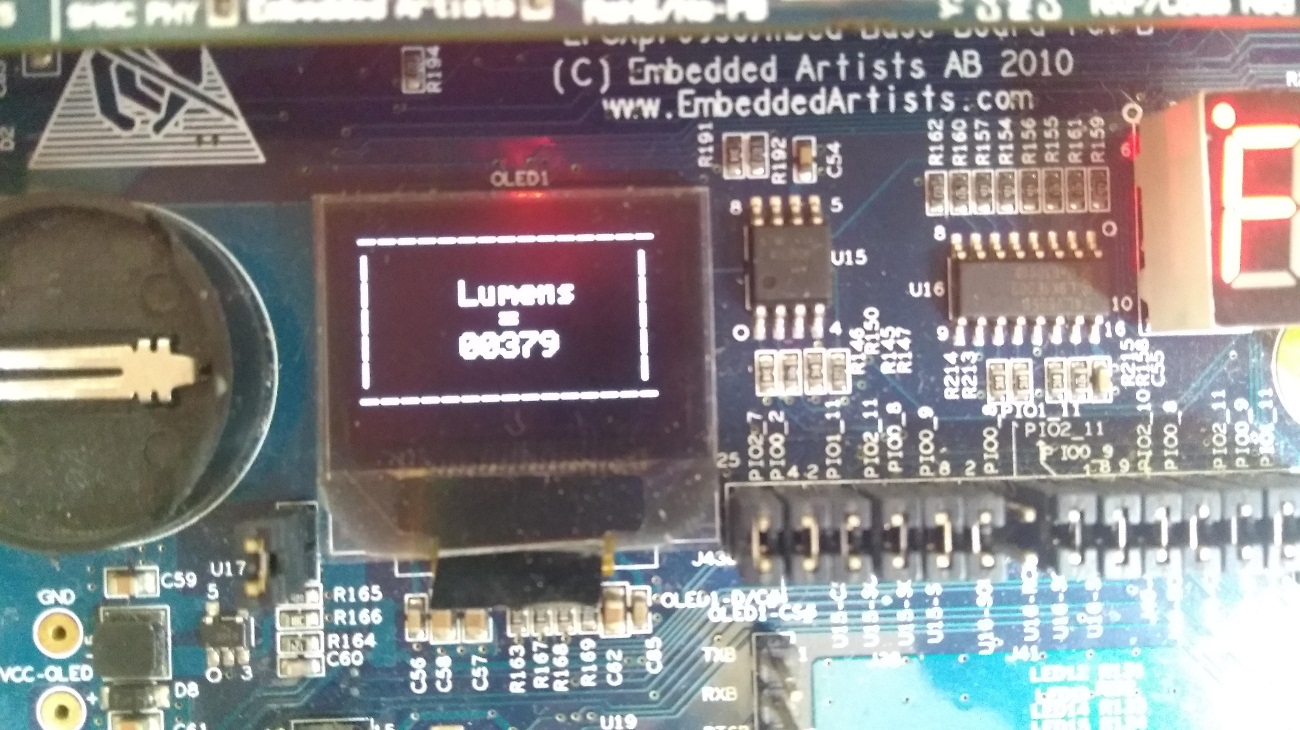
Para alterar o sensor que está sendo lido basta clica no *button*, mudando para modo de exibição: inicial, dados de luminosidade e de temperatura.

A Figura 21 mostra os nomes dos integrantes do grupo.



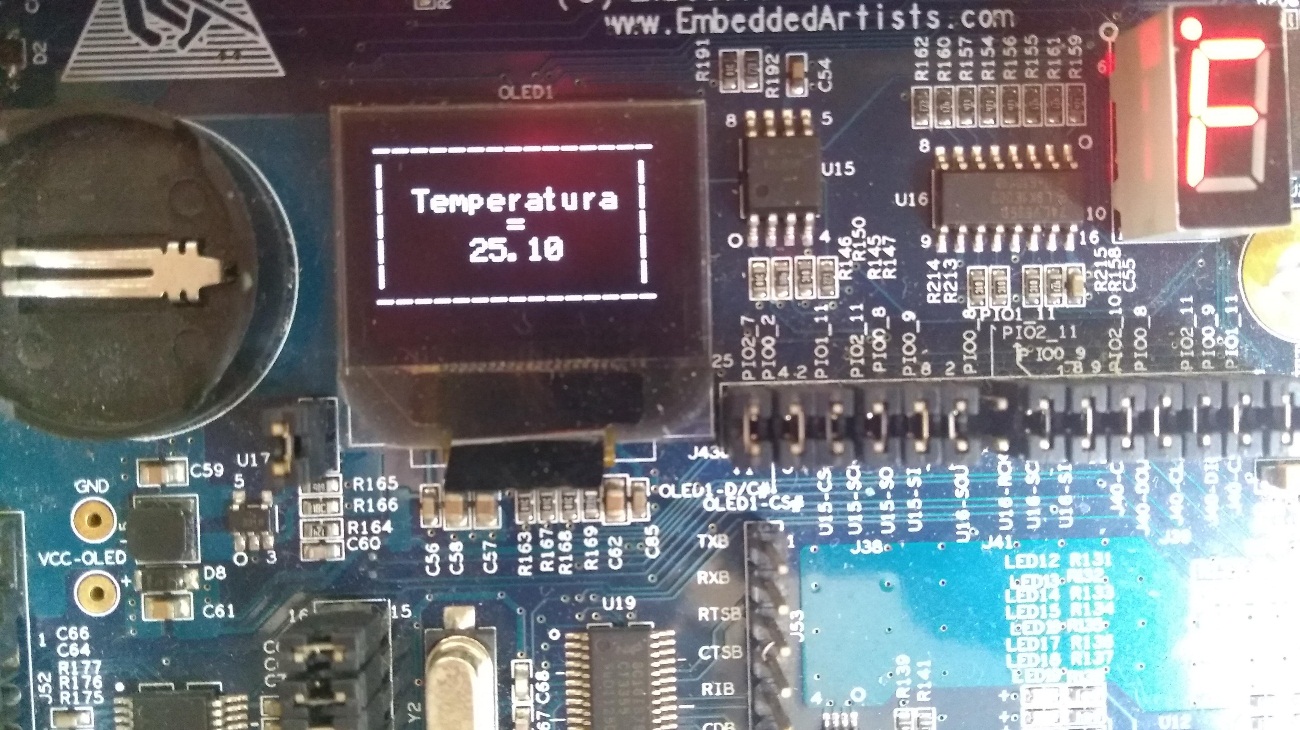
*Figura 21 - Resultado da leitura do sensor e estado da válvula aberta*

A Figura 22 mostra o sistema em funcionamento com o *display* exibindo a luminosidade atual.



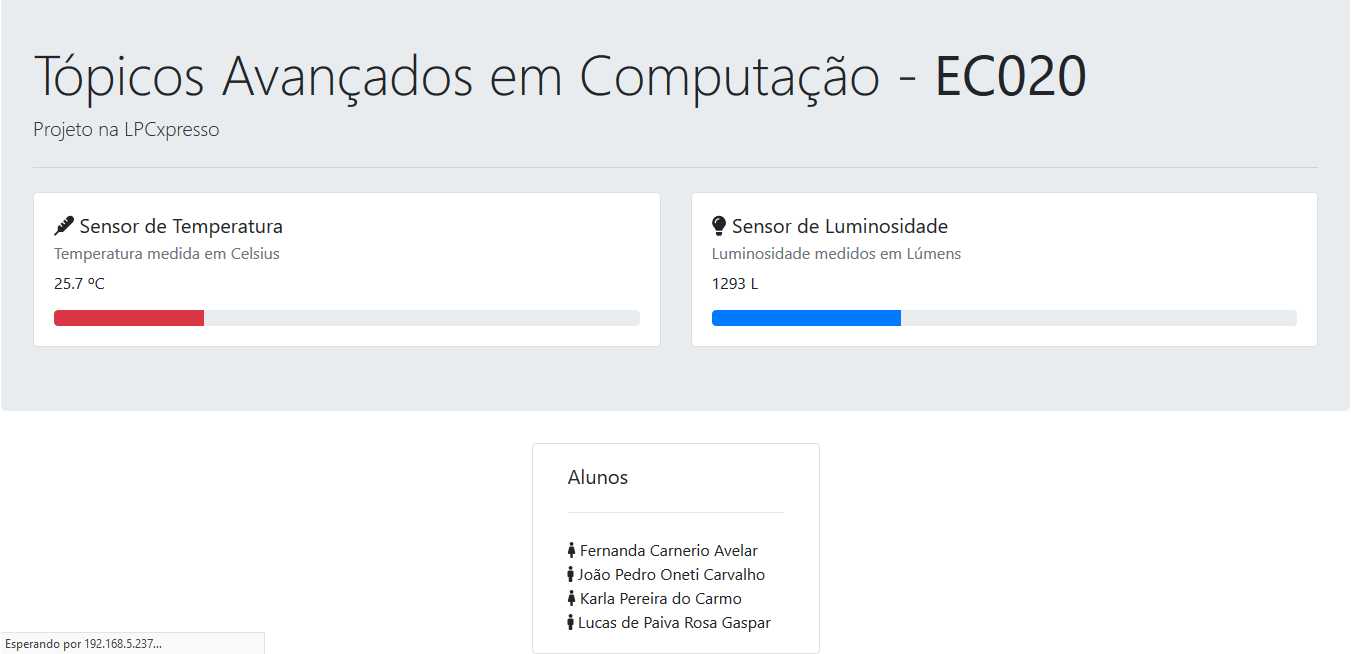
*Figura 22 - Resultado da leitura do sensor de luminosidade medidos em Lumens*

A Figura 23 mostra o sistema em funcionamento com o *display* exibindo a temperatura atual.



*Figura 23 - Resultado da leitura do sensor de temperatura medidos em Celsius*

A Figura 25 mostra a interface ethernet exibindo os valores de temperatura e de luminosidade enviados pelo UART.

****

*Figura 25 - Temperatura e luminosidade exibidas na interface web.*

**VIII. CONCLUSÕES**

O RTOS auxilia a simplificar o esforço complexo de integração em sistemas de arquivos, componentes USB, TCP/IP e GUI (*easyweb*), ao contrário do bare-metal que exige mais esforço para a implementação.

Um RTOS é projetado para permitir que cada tarefa seja fornecida com uma prioridade que ajuda a garantir que a tarefa seja executada de maneira determinística. A configurabilidade do RTOS permite que um desenvolvedor habilite os recursos necessários para o aplicativo enquanto desabilita recursos que, de outra forma, ocupariam muita memória.

Muitos RTOSs populares são desenvolvidos para que possam ser facilmente movidos de uma arquitetura de microcontrolador para a próxima. Um RTOS disponibiliza recursos para ser usados pelo desenvolvedor, com uma maneira de criar tarefas, destruir tarefas, tarefas síncronas, comunicar-se entre tarefas e bloquear recursos, ao contrário do bare-metal que é preciso que o desenvolvedor faça essa recriação.

Sendo assim, conclui-se que para esse projeto, o uso de um RTOS, no caso do nosso projeto, FreeRTOS, tornou-se mais simples e mais eficiente, portando como um processo ágil, ao contrário da implementação com bare-metal.

**IX. REFERÊNCIAS**

[1]

FRIGIERI, E. P., D’ÂNGELO, V. I., RAMOS, R. F. M., JUNIOR, L. C. A., TEIXEIRA, P. S. F. L. Portando Uma Aplicação de Sistema Embarcado Com Arquitetura Super Loop Para Operar Com Sistema Operacional De Tempo Real. IFMG campos Formiga, v. 2, n. 1, p. 47-67, jan./jun. 2014.

[2]

Link do projeto no GitHub: https://github.com/jponeticarvalho/EC020

[3]

EMBEDDED ARTISTS AB. *LPCXpresso Base Board: User’s Guide.* Malmo, Suécia, 2010. PDF. 13 Fevereiro 2017.

[4]

Embarcados, Comunicação entre tarefas no FreeRTOS: Filas. Disponível em:

< https://www.embarcados.com.br/comunicacao-entre-tarefas-no-freertos-filas/> Acesso em: 17 de Junho de 2018.

[5]

# Embedded Related, Jacob Beningo, From bare-metal to RTOS. Disponível em:

<https://www.embeddedrelated.com/showarticle/1008.php>. Acesso em: 17 de Junho de 2018.