# **Módulos Arduino Nano**

1. Setup
   * Responsável: Guillermo - Pronto!
   * Executada apenas uma vez na ligação do respirador.
   * Configuração dos pinos E/S, interrupções e inicialização de variáveis.
2. Timer 0
   * Responsável: Guillermo - Pronto!
   * Contagem do tempo das tarefas com resolução de 1 ms.
   * Dispara outras tarefas como debouncing, medição, .....
   * Controle da rampa de aceleração ou desaceleração do motor (CC ou passo)
3. Timer 1 ou PWM
   * Responsável: Guillermo - Pronto!
   * Controle da velocidade do motor (CC ou passo)
   * Contagem da quantidade de passos do motor de passo.
4. Medição
   * Responsável: Guillermo - Pronto!
   * Obtenção das medições do ADC.
   * Dispara início da medição na interrupção do Timer0 (cada 1 ms) de forma automática . O seja, cada 4 ms temos as 4 medições do ADC.
   * Na interrupção de fim de conversão, são obtidas as medições em sequência.
   * O processo é contínuo, não sendo necessário iniciar a conversão. O processamento pode ser feito em qualquer momento
   * É recomendável fazer uma cópia temporária do valor a ser processado se a mudança do valor durante o processamento (quando acontecer uma interrupção do ADC) pode afetar o cálculo.
5. Processamento das medições
   * Responsável: Gabriel Williams
   * Função Independiente que vai ser chamada no loop principal quando MeasureState = MPROCESSING
   * Leitura de valores de calibração da EEPROM (podem ser lidos no setup e armazenados numa variável).
   * Ajuste de offset, ganho e conversão a unidades de engenheira de cada medição.
   * Uso de inteiros 16 bits com aritmética de ponto fixo se possível.
6. Atendimento ao teclado
   * Responsável: Guillermo - Pronto!
   * Usar interrupções de mudança de porta - Pin Change Interrupt)
   * Pinos D4...D7 (PD4...PD7) vão ativar interrupções PCINT20...23 correspondentes ao vetor de interrupção PCI2
   * Table of pins > Pin change names / masks
   * D4 PCINT20 (PCMSK2 / PCIF2 / PCIE2)
   * D5 PCINT21 (PCMSK2 / PCIF2 / PCIE2)
   * D6 PCINT22 (PCMSK2 / PCIF2 / PCIE2)
   * D7 PCINT23 (PCMSK2 / PCIF2 / PCIE2)
   * Dependendo do estado, pode disparar outros processos usando bandeiras.
7. LCD
   * Responsável: Programador 2
   * Comunicação SPI como o LCD Gráfico.
   * Função disparada pelo teclado e chamada no loop principal.
   * Representação de menus para configuração.
   * Usa máquina de estados, que vai mudar dependendo das flags do teclado.
   * Escritura em EEPROM dos valores de configuração.
   * Usar bibliotecas U8glib2 (<https://github.com/olikraus/u8g2/wiki>) ou U8glib1.19.1 (<https://github.com/olikraus/u8glib>) e LCDMenuLib2 (<https://github.com/Jomelo/LCDMenuLib2>)

Exemplos e tutoriais:

<https://github.com/olikraus/U8glib_Arduino>

<https://www.dobitaobyte.com.br/display-128x64-com-arduino-utilizando-u8g2-lib/>

<http://henrysbench.capnfatz.com/henrys-bench/u8glib-graphics-library-user-guide/u8glib-arduino-oled-tutorial-2-playing-with-the-picture-loop/>

<https://www.youtube.com/watch?v=CxFyXVbHRU0>

<https://www.arduinoecia.com.br/display-grafico-lcd-128x64-st7920-arduino/>

<https://www.arduinoecia.com.br/relogio-arduino-display-128x64-rtc-ds3231/>

<http://henrysbench.capnfatz.com/henrys-bench/u8glib-graphics-library-user-guide/u8glib-arduino-oled-tutorial-1-hello-world-on-steroids/>

1. Início de curso
   * Responsável: Guillermo - Pronto!
   * Zera posição, reinicia o ciclo respiratório
   * Por enquete em cada passo do motor de passo ou interrupção externa 1 por borda se usado motor de CC.
2. Tacômetro (opcional, interrupção externa 0 por borda)
   * Responsável: Guillermo
   * Medição da velocidade e avanço do motor de CC.
3. Processamento de alarmes (chamado no loop principal)
   * Responsável: Gabriel Williams
   * Comparação de valores das medições com valores limites.
   * Leitura de valores limites de alarme da EEPROM.
   * Disparo do alarme no buzzer (ativa PWM).
   * Para o reconhecimento do alarme, verifica no estado do teclado se foi apertada alguma tecla)
4. Controle do ciclo respiratório
   * Responsável: Programador 3
   * Função chamada no loop principal ou no Timer0 (vamos avaliar depois)
   * Mudança dos parâmetros do motor dependendo da configuração, com uma máquina de estados finitos.
5. Comunicação via serial para monitoramento desde uma PC (usar USART, não é prioridade, atendimento no loop principal)
   * Responsável: Guillermo
   * Roda no loop principal (não é prioridade)
   * UART hard
6. Comunicação via serial com a ESP01. Solicitação de valores de configuração
   * Responsável: Guillermo
   * Disparada pelo Timer0 cada 100 ms quando ativa flag
   * Envia várias medições por vez (criar uma lista circular com ponteiro)
   * Programação da ESP-01 (desligar aparelho, tirar módulo bluetooth, jumpear pinos 3-4 do J5, ligar aparelho e enviar arquivo da ESP-01)
   * Roda no loop principal (não é prioridade)
   * UART soft
7. Buzzer
   * Responsável: Guillermo - Pronto!
   * Gerar o som em diferentes frequências.

# **Módulos ESP01**

1. Recebimento de medições e configuração do Arduino. Envio de valores de configuração.
   1. Responsável: Felipe Sobrinho
   2. Disparada pela solicitação do Arduino.
   3. Recebe várias medições por vez
   4. Atendimento por interrupção de Rx
   5. Uso de UART hard.
2. Comunicação com módulo bluetooth.
   1. Responsável: Felipe Sobrinho
   2. Envio de medições e configuração Recebimento de valores de configuração.
   3. Uso de UART soft.

<http://myosuploads3.banggood.com/products/20190129/20190129043725SKUA87502.pdf>

<https://cadernodelaboratorio.com.br/2017/04/08/iniciando-com-o-arduino-modulo-bluetooth-i/>

<http://mundoprojetado.com.br/modulo-bluetooth-comunicando-c-arduino-parte-1/>

1. Programação por via serial ou OTA se possível.
   1. Responsável: Felipe Sobrinho