Universidade Federal do ABC

Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas

PPG EBM107 – Tópicos Avançados em Instrumentação Biomédica
2º quadrimestre de 2019

Roteiro 05:
Medição de sinais com Arduino

1 Objetivo

Nesta aula prática iremos construir os circuitos de condicionamento de sinal estudados para que seja possível utilizar o conversor AD no Arduino para digitalizar sinais AC.

2 Introdução

No roteiro anterior vimos como medir sinais analógicos no Arduino usando a função `AnalogRead()`. Apesar de prático, o uso dessa função permite apenas taxas de amostragem menores que $10kHz$, uma vez que a função não é otimizada para ler diversos valores em sequência, e leva cerca de $100\mu s$ para rodar.

O Arduino possui um modo de coleta rápida chamado *ADC Free Running mode*, no qual o ADC converte, continuamente, sinais analógicos, disparando uma interrupção ao final de cada conversão. Assim, não é desperdiçado tempo entre o final de uma coleta e o início de outra.

Para entender a taxa de amostragem do Arduino, devemos lembrar que o *clock* do ADC é controlado por um fator de pré-escala do *clock* do Arduino, ou seja, o *clock* do ADC funciona a uma fração do *clock* do Arduino, sendo essa fração ajustada como uma divisão por um número inteiro entre 2 e 128. Assim, como o *clock* do Arduino Uno funciona a $16MHz$, o *clock* do ADC funciona entre $8MHz$ e $125kHz$, dependendo do fator ajustado. Como o Arduino precisa de 13 ciclos do *clock* para realizar uma medição, temos uma taxa de amostragem teórica entre $615kHz$ e $9,6kHz$.

O fator de pré-escala é ajustado através de 3 bits: `ADPS2`, `ADPS1` e `ADPS0`, conforme a tabela a seguir.

Pré-escala	`ADPS2`	`ADPS1`	`ADPS0`	*Clock* do ADC (MHz)	Taxa de amostragem (kHz)
2	0	0	1	8	615
4	0	1	0	4	307
8	0	1	1	2	153
16	1	0	0	1	76,8
32	1	0	1	0,5	38,4
64	1	1	0	0,25	19,2
128	1	1	1	0,125	9,6

Tabela 1: Taxas de amostragem para o ADC do Arduino

Um programa que mostra o funcionamento desse modo pode ser visto a seguir.

Listing 1: calcula-taxa.ino

```
1 /*************************************************************************
2  * Coleta 1000 pontos no ADC, calcula a taxa de amostragem e imprime os
3  * resultados na serial.
4  * Codigo baseado no exemplo disponibilizado no site:
5  * http://yaab-arduino.blogspot.com/2015/02/fast-sampling-from-analog-input.html
```

```
 ********************************************************************************/
int numSamples=0;
long t, t0;
byte x;

void setup()
{
  Serial.begin(115200);

  ADCSRA = 0;             // clear ADCSRA register
  ADCSRB = 0;             // clear ADCSRB register
  ADMUX |= (0 & 0x07);    // set A0 analog input pin
  ADMUX |= (1 << REFS0);  // set reference voltage
  ADMUX |= (1 << ADLAR);  // left align ADC value to 8 bits from ADCH register

  // sampling rate is [clock] / [prescaler] / [conversion clock cycles]
  // for Arduino Uno clock is 16 MHz and a conversion takes 13 clock cycles
  //ADCSRA |= (1 << ADPS2) | (1 << ADPS0);    // 32 prescaler for 38.5 KHz
  //ADCSRA |= (1 << ADPS2);                   // 16 prescaler for 76.9 KHz
  ADCSRA |= (1 << ADPS1) | (1 << ADPS0);    // 8 prescaler for 153.8 KHz
  //ADCSRA |= (1 << ADPS1);                   // 4 prescaler for 307 KHz
  //ADCSRA |= (1 << ADPS0);                   // 2 prescaler for 615 KHz

  ADCSRA |= (1 << ADATE); // enable auto trigger
  ADCSRA |= (1 << ADIE);  // enable interrupts when measurement complete
  ADCSRA |= (1 << ADEN);  // enable ADC
  ADCSRA |= (1 << ADSC);  // start ADC measurements
}

ISR(ADC_vect)
{
  x = ADCH;  // read 8 bit value from ADC
  numSamples++;
}

void loop()
{
  if (numSamples>=1000)
  {
    t = micros()-t0;  // calculate elapsed time

    Serial.print("Frequencia de amostragem: ");
    Serial.print((float)1000000/t);
    Serial.print(" KHz. Valor lido: ");
    Serial.println(x);
    delay(2000);

    // restart
    t0 = micros();
    numSamples=0;
  }
}
```

Para verificar se o programa está funcionando corretamente, ligue um potenciômetro ao Arduino, conforme figura 1.

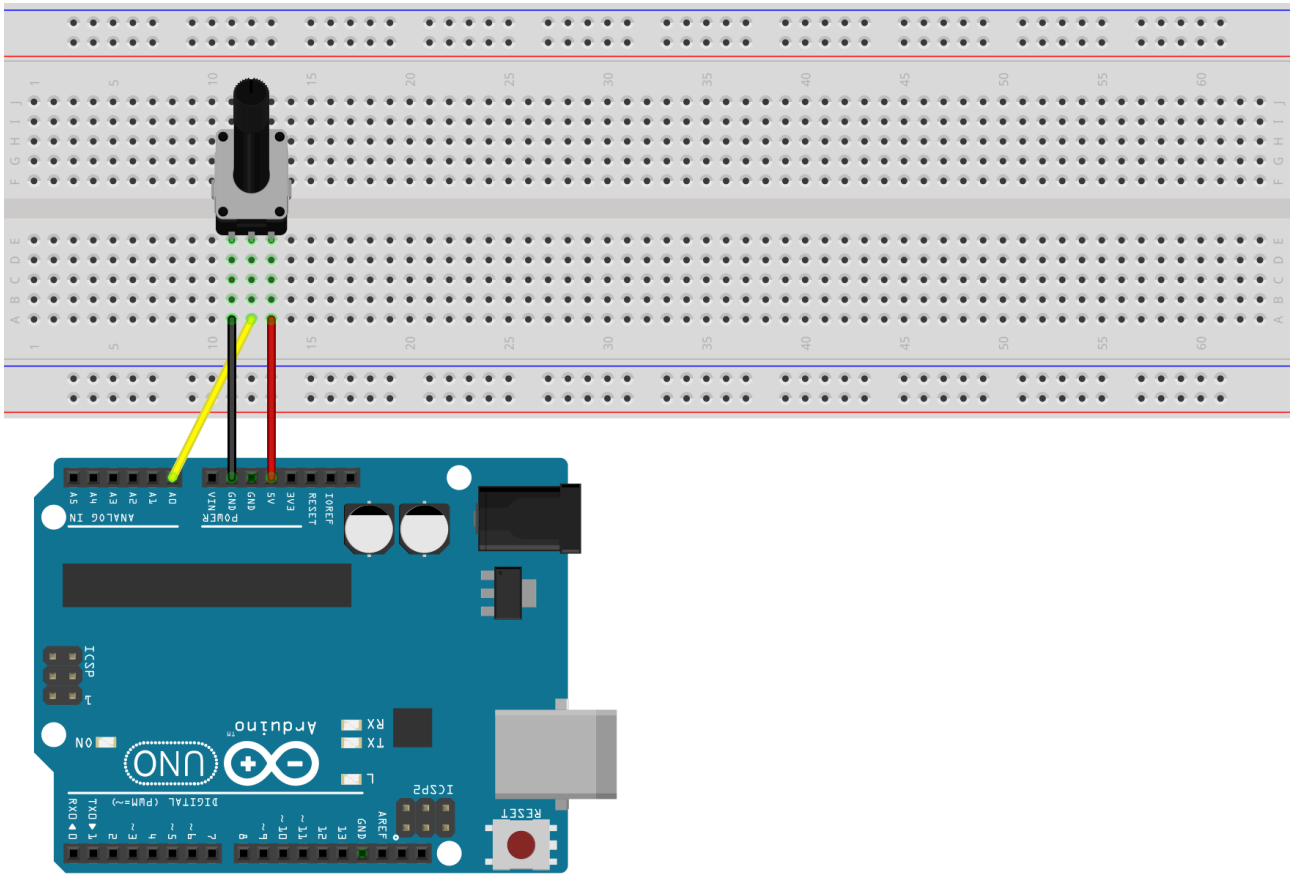

Figura 1: Conectando um potenciômetro a uma entrada analógica do Arduino Uno.

Para o relatório

Grave o programa de exemplo no seu Arduino, e verifique a taxa de amostragem real para cada fator da tabela 1. Para isso, (des)comente (ou escreva) a linha correspondente aos bits `ADPS2`, `ADPS1` e `ADPS0` do fator desejado. Verifique, para cada escala, se os valores medidos estão coerentes com a posição do potenciômetro, variando a posição do mesmo. Lembre-se que este programa usa apenas 8-bits do ADC, logo os valores medidos variam de 0 a 255. Monte uma tabela com os resultados, indicando quais taxas de amostragem funcionaram como esperado.

3 Soldando a placa de condicionamento de sinais

Agora iremos soldar a placa de condicionamento de sinais. O esquema elétrico da placa, assim como os valores dos componentes que devem ser utilizados, podem ser vistos na figura 2.

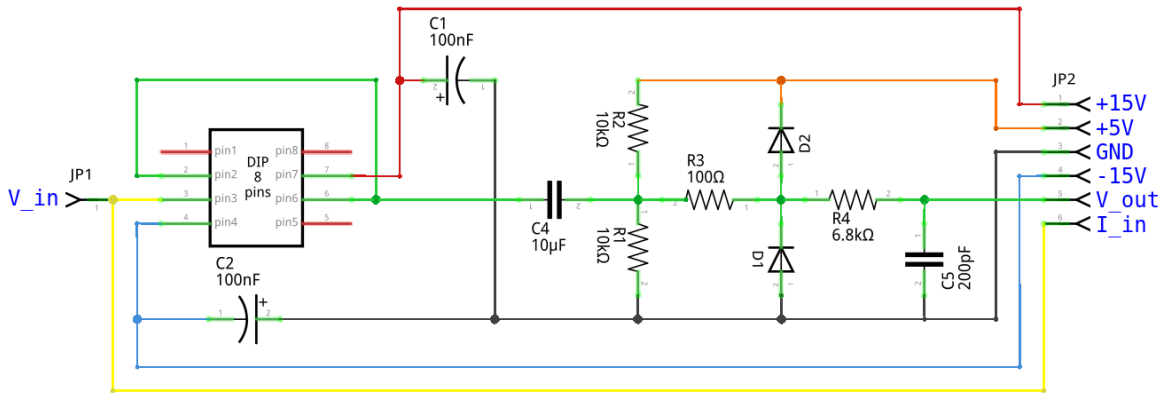

Figura 2: Esquema elétrico da placa de condicionamento de sinais.

Para padronizar o tamanho das placas, vocês devem soldar as trilhas e posicionar os componentes na placa conforme a figura 3.

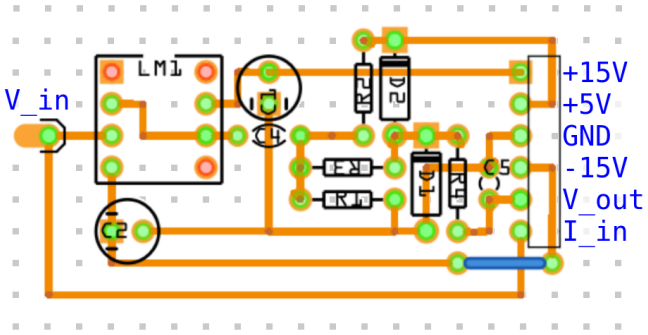

Figura 3: Diagrama das soldas da placa de condicionamento de sinais usando uma placa de fenolite furada.

A figura 4 mostra como devem ficar as soldas, quando a placa é vista por baixo.

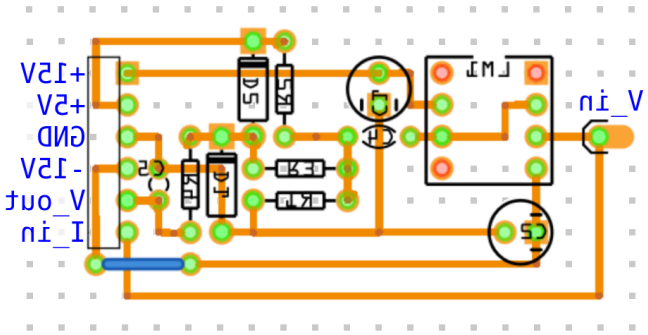

Figura 4: Diagrama espelhado das soldas da placa de condicionamento de sinais usando uma placa de fenolite furada.

Para o relatório

Cada grupo deverá soldar 3 placas de condicionamento de sinal, e incluir uma foto dos dois lados das placas soldadas no relatório. Após soldar as placas, vocês devem, com o auxílio de um gerador de ondas e de um osciloscópio, verificar se estas estão funcionando corretamente, ou seja, se o sinal na saída apresenta o *offset* de $2,5V$, se o sinal a $25kHz$ não sofre alteração de amplitude, e se sinais acima de $50kHz$ são atenuados. Apresentem os resultados e os gráficos obtidos, para cada placa, no relatório.

Referências Bibliográficas

- MOLLOY D., Exploring Raspberry Pi: Interfacing to the real world with embedded linux, Wiley,2016.
- Fast sampling from analog input (http://yaab-arduino.blogspot.com/2015/02/fast-sampling-from-analog-input.html). Acesso em Agosto de 2019.