Guia de Trabalho Final: Aquisição e Processamento de Sinais com Arduino

Prof. Dr. William Humberto Cuéllar Sánchez
Centro Universitário UDF

22 de Novembro de 2024

Objetivo

Este experimento visa introduzir os alunos à aquisição e ao processamento de sinais utilizando um sensor, filtros analógicos e um microcontrolador Arduino. Os alunos deverão utilizar um sensor para coletar dados, aplicar filtros analógicos para melhorar o sinal e, finalmente, processar o sinal no Arduino utilizando filtros digitais. O trabalho também envolve o uso de amplificadores para ajustar o sinal do sensor, garantindo que ele esteja em um nível adequado para ser lido pelo microcontrolador.

Material Necessário

- Arduino (Uno ou similar)
- Protoboard e jumpers
- Sensor (escolha entre sensor de temperatura LM35, sensor de luminosidade LDR, sensor de pressão MPX5010, potenciômetro, ou sensor de umidade de solo)
- Componentes passivos (resistores e capacitores)
- Amplificador operacional (por exemplo, LM358)
- Fonte de alimentação
- Multímetro

Parte 1: Aquisição de Dados com Sensor

Escolha um dos sensores abaixo para a aquisição dos dados:

- **LM35**: Sensor de temperatura analógico.
- **LDR**: Sensor de luminosidade que varia a resistência de acordo com a intensidade da luz.
- **MPX5010**: Sensor de pressão analógico que fornece uma tensão proporcional à pressão medida.
- **Potenciômetro**: Utilizado como um sensor analógico de posição ou ajuste de tensão.
- **Sensor de umidade de solo**: Sensor que fornece uma leitura analógica proporcional à umidade do solo.

Conecte o sensor escolhido à protoboard e faça as conexões necessárias para que o Arduino possa ler os dados do sensor. Verifique no datasheet do sensor como realizar as conexões adequadas. Primeiro, teste o sensor diretamente com o Arduino, garantindo que todas as leituras estejam corretas. Em seguida, desconecte o sinal do sensor do Arduino e conecte-o à entrada do filtro com amplificadores criados para filtrar o sinal antes de retornar ao Arduino. Nas entradas do amplificador, utilize 0V e 5V como referência, garantindo que o amplificador escolhido permita uma fonte de alimentação normal, sem necessidade de fonte simétrica.

Parte 2: Filtros Analógicos e Amplificadores

Você deverá utilizar pelo menos um dos dois tipos de filtros analógicos a seguir:

1. **Filtro Passa-Baixas**: O objetivo é atenuar ruídos de alta frequência que possam estar presentes no sinal do sensor. Um exemplo é o uso de um filtro RC (Resistor-Capacitor) para suavizar a leitura de um sensor de temperatura LM35.

2. **Filtro Passa-Altas com Amplificador**: Utilize um amplificador operacional para amplificar sinais de baixa intensidade. Um exemplo da aplicação deste filtro é no uso de um sensor de vibração, onde você deseja atenuar ruídos de baixa frequência que não representam a vibração relevante para a medição. Utilize um circuito com amplificador operacional (como o LM358) para ajustar o ganho do sinal e garantir que ele esteja em um nível apropriado para o Arduino.

Monte o filtro escolhido, conecte-o ao circuito e analise o sinal de saída com o multímetro para confirmar se está adequado ao nível de entrada do Arduino (entre 0 e 5V). O sinal de entrada do amplificador deve ser o sinal do sensor, e a saída do filtro deve ser conectada ao Arduino. Lembre-se de que todas as terras (do sensor, do amplificador e do Arduino) devem estar interconectadas.

Parte 3: Processamento de Sinal no Arduino

O próximo passo é conectar a saída do circuito filtrado a uma entrada analógica do Arduino. O sinal será lido e processado internamente usando um dos seguintes filtros digitais:

1. **Filtro de Média Móvel**: Utilize a média de um número fixo de amostras passadas para suavizar o sinal recebido. Esse filtro é eficaz na redução de ruídos de alta frequência.

2. **Filtro Mediana**: Esse filtro pode ser usado para atenuar ruídos do tipo "impulso", onde picos esporádicos afetam a leitura. Ele consiste em calcular a mediana de um conjunto de amostras.

3. **Filtro Gaussiano**: Pode ser implementado como um tipo de média ponderada das amostras, dando maior peso aos valores centrais, resultando em uma suavização mais eficiente em alguns casos.

Escreva um código Arduino para ler o valor do sensor e aplicar um dos filtros digitais sugeridos. O código deve imprimir o valor filtrado no monitor serial.

Instruções

1. Conecte todo o circuito e verifique cada uma das conexões antes de ligar a fonte de alimentação.

2. Teste o sinal no Arduino sem nenhum filtro e depois implemente o filtro analógico e digital.

3. Compare os sinais com e sem filtro e anote as observações sobre as diferenças percebidas.

4. Documente seu trabalho explicando o processo de montagem, os valores utilizados para cada componente do filtro e o código implementado.

Resultados Esperados

Os alunos deverão apresentar os resultados das leituras do sensor sem e com filtros analógicos. Deverão incluir uma análise da melhora na qualidade do sinal após a aplicação dos filtros e descrever situações práticas onde esses tipos de filtros são necessários para um bom funcionamento do sistema.

Entrega

- Montagem do sistema, os circuitos utilizados e o código do Arduino.
- Data de entrega: 06/12/2024.

1 Exemplo Guia: Ajuste do Sensor LM35 e Escolha dos Filtros

Este é um exemplo do que você pode realizar, aqui não coloquie a parte do filtro com amplificador demaneira completa e só comentei a melhor opção (Não quis colocar o desenho eletrônico de como implementar o fitro)

1.1 Ajuste do Sensor LM35

Para garantir a precisão das medições do sensor LM35, é importante realizar um ajuste inicial utilizando duas temperaturas conhecidas. O processo é o seguinte:

1.1.1 Água Gelada ($0°C$)

Coloque o sensor LM35 em contato com água gelada (pode-se usar um recipiente com gelo). Aguarde alguns minutos até que o sensor estabilize e registre a leitura exibida pelo Arduino. Esta leitura deve ser ajustada para corresponder a $0°C$. Caso o valor não esteja próximo de $0°C$, anote a diferença para correção futura, chamando-a de V_0.

1.1.2 Temperatura Ambiente

Meça a temperatura do ambiente usando um termômetro confiável. Coloque o sensor LM35 em contato com o ar e aguarde até que a leitura se estabilize. Compare a leitura do Arduino com a do termômetro e anote a temperatura ambiente T_{amb} e a leitura correspondente V_{amb}. Ao medir uma nova temperatura, você obterá um novo valor V. Como V é um valor entre 0 e 1023, você pode criar uma equação linear para medir a temperatura, dada por:

$$T_{med} = T_0 + \frac{(T_{amb} - T_0)}{(V_{amb} - V_0)} \cdot (V - V_0),$$

ou

$$T_{med} = \frac{(T_{amb} - T_0)}{(V_{amb} - V_0)} \cdot (V - V_0),$$

onde os termos são definidos conforme descrito anteriormente.

No Arduino, a leitura do sensor V é um valor digital entre 0 e 1023.

Essa equação pode ser implementada no Arduino para calcular a temperatura ajustada a partir do valor de leitura digital V.

1.2 Escolha do Filtro Analógico

Para este experimento, foi escolhido um filtro passa-baixas. Esse tipo de filtro é ideal para sensores de temperatura como o LM35, pois ajuda a atenuar ruídos de alta frequência, que podem ser causados por interferências externas ou flutuações rápidas não representativas da variação real de temperatura. Como a temperatura tende a mudar de forma relativamente lenta, faz sentido eliminar variações rápidas e focar apenas na tendência principal do sinal, garantindo leituras mais estáveis.

1.3 Filtro Mediana para Processamento Digital

Após o ajuste do sinal e a aplicação do filtro analógico, utilizamos um filtro mediana de janela 3 para o processamento digital. O filtro mediana é útil para remover picos esporádicos (ruídos tipo "impulso") que possam ocorrer devido a interferências ocasionais. O funcionamento é simples: coletamos 3 amostras consecutivas, ordenamos os valores e selecionamos o valor do meio. Isso garante que picos anômalos não influenciem significativamente o valor final, proporcionando uma leitura mais confiável e robusta, especialmente em ambientes onde o ruído elétrico pode afetar as medições.

1.4 Codigo Arduino para Ajuste do Sensor LM35

```
1    // Codigo Arduino para ajustar o sensor LM35
2
3    const int sensorPin = A0; // Pino do sensor LM35
4    float temperatura;
```

```
5
6     // Constantes para ajuste
7     const float T0 = 0.0;        // Temperatura de referencia (0 graus)
8     const float V0 = 25.0;       // Leitura digital correspondente a T0
9     const float Tamb = 28.0;     // Temperatura ambiente conhecida
10    const float Vamb = 143.0;    // Leitura digital correspondente a Tamb
11
12    void setup() {
13      Serial.begin(9600); // Inicializa a comunicacao serial
14    }
15
16    void loop() {
17      int V = analogRead(sensorPin); // Leitura do valor digital do sensor
18
19      // Calculo da temperatura ajustada
20      temperatura = ((Tamb - T0) / (Vamb - V0)) * (V - V0);
21
22      // Exibe a temperatura calculada
23      Serial.print("Temperatura: ");
24      Serial.print(temperatura);
25      Serial.println(" graus C");
26
27      delay(1000); // Atraso de 1 segundo entre leituras
28    }
```

Listing 1: Código Arduino LM35 com ajuste

1.5 Código Arduino para Calcular a Mediana com Ajuste do Sensor LM35

```
1     const int sensorPin = A0; // Pino do sensor LM35
2     float valores[3];
3     const float V0 = 25.0; // Valor de leitura para T0 = 0 C
4     const float Tamb = 28.0; // Temperatura ambiente conhecida
5     const float Vamb = 143.0; // Valor de leitura para Tamb (obtido
          experimentalmente)
6
7     void setup() {
8       Serial.begin(9600); // Inicializa a comunicacao serial
9     }
10
11    void loop() {
12      // Leitura dos 3 valores
13      for (int i = 0; i < 3; i++) {
14        int V = analogRead(sensorPin); // Leitura direta do valor digital do
              sensor
15        valores[i] = ((Tamb - 0) / (Vamb - V0)) * (V - V0); // Calculo da
              temperatura ajustada
16        delay(500); // Atraso para cada leitura
17      }
18
19      // Ordenacao dos valores (simples bubble sort para 3 elementos)
20      for (int i = 0; i < 2; i++) {
21        for (int j = i + 1; j < 3; j++) {
22          if (valores[i] > valores[j]) {
23            float temp = valores[i];
24            valores[i] = valores[j];
25            valores[j] = temp;
26          }
27        }
28      }
29
30      // Exibir o valor da mediana
```

```
31        Serial.print("Valor da Mediana: ");
32        Serial.print(valores[1]); // O valor do meio apos ordenacao
33        Serial.println(" C");
34
35        delay(2000); // Atraso de 2 segundos entre calculos de mediana
36      }
```

Listing 2: Código Arduino para calcular a mediana de 3 valores lidos do LM35 com ajuste