UNIVERSIDADE FEDERAL DE UBERLÂNDIA - UFU

FACULDADE DE ENGENHARIA ELÉTRICA – FEELT

SINAIS E SISTEMAS EM ENGENHARIA BIOMÉDICA

Utilização das portas PWM do Arduino

Alunos:

1. Ítalo Gustavo Sampaio Fernandes - 11511EBI004
2. Nathalia Rodrigues - 11411EBI018
3. Paulo Camargos Silva - 11611EBI023

Prof. Sérgio Ricardo de Jesus Oliveira

Uberlândia, **4** de **outubro** de 2017

Utilização das portas analógicas do Arduino

1 – Introdução

O Arduino DUE possui 54 pinos digitais de entrada ou saída, onde 12 desses podem ser utilizados para controle PWM aumentando a capacidade de controle PWM se comparada a Arduino UNO que possui apenas 6 saídas PWM. Os conversores analógicos-digital (ADC) dessa placa são rápidos e podem operar em até 1 MHz de Frequência de amostragem.

Essa nova placa possui 12 entradas de ADC, com 12 bits de resolução. Além disso, a placa possui dois conversores digital-analógicos (DAC) de 12 bits, o que permite mais versatilidade a diferentes aplicações. É possível então, ter uma saída com um valor de tensão entre 0 e 3,3V, com 4096 diferentes valores, nesse intervalo. Pela existência destes conversores D/A, é possível gerar sinais senoidais, triangulares, quadradas, dente-de-serra e etc., com diversas frequências, de acordo com o período da onda analógica em questão.

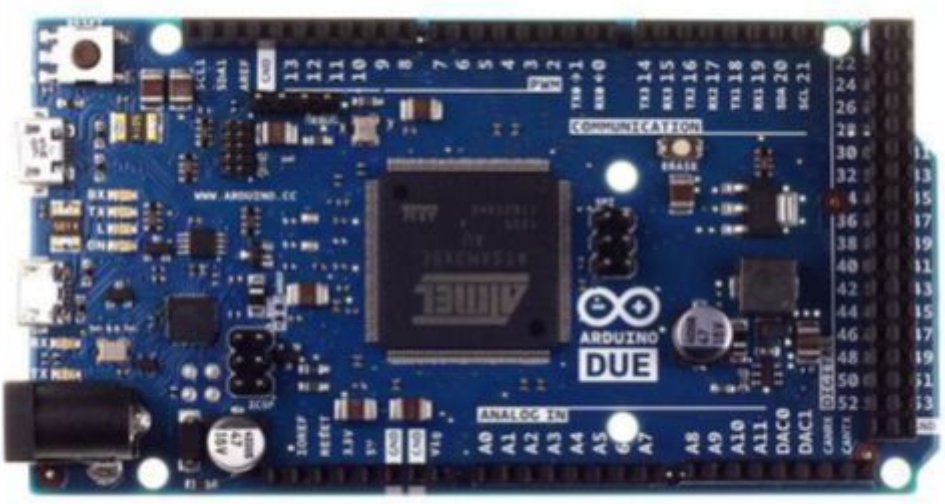

Figura 1 - Arduino DUE

Um motor CC nada mais é do que um motor alimentado por corrente contínua, sendo esta alimentação proveniente de uma bateria ou qualquer outra de alimentação CC. O motor cc pode ser controlado apenas variando a sua tensão, diferentemente de um motor elétrico de corrente alternada (CA) cuja velocidade é variada pela frequência. Um motor CC é composto por um eixo acoplado ao rotor que é a parte girante do motor. Na Figura 1, o estator é composto por um ímã e o comutador tem a função de transferir a energia da fonte de alimentação ao rotor. Na Figura 1 é também é possível observar as partes que compõem um motor CC.

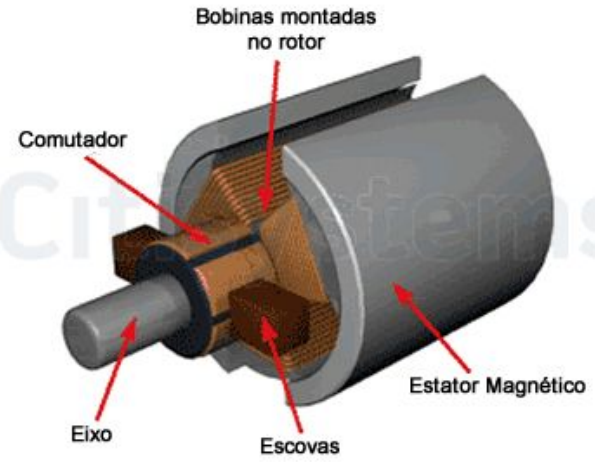

Figura 2 – Motor *CC* e suas partes.

O funcionamento dos motores CC baseia-se no princípio do eletromagnetismo clássico pelo qual um condutor carregando uma corrente e mergulhado em um fluxo magnético fica submetido a uma força eletromagnética. Energia elétrica é fornecida aos condutores do enrolamento da armadura pela aplicação de uma tensão elétrica em seus terminais pelo anel comutador, fazendo com que se circule uma corrente elétrica nesse enrolamento que produz um campo magnético no enrolamento da armadura.

Como o corpo do estator é constituído de materiais ferromagnéticos, ao aplicarmos tensão nos terminais do enrolamento de campo da máquina temos uma intensificação do campos magnéticos no mesmo e, portanto, a produção de pólos magnéticos (Norte e Sul) espalhados por toda a extensão do estator. Quando aplicamos uma tensão no comutador, com a máquina parada, a tensão é transferida ao enrolamento da armadura fazendo com que circule uma corrente pelo mesmo, o que produz um campo magnético e outros pares de pólos no enrolamento da armadura. A orientação desse campo permanece fixa, simultaneamente temos uma tensão elétrica aplicada no enrolamento de campo no estator, assim, ao termos a interação entre os campos magnéticos da armadura no rotor e do campo no estator, os mesmos tentarão se alinhar.

Assim surgirá um binário de forças que produzirá um torque no eixo, fazendo o mesmo girar. Ao girar, o eixo gira o anel comutador que é montado sobre o eixo, e ao girar o anel comutador muda o sentido de aplicação da tensão, o que faz com que a corrente circule no sentido contrário, mudando o sentido do campo magnético produzido. No entanto, ao girar o anel comutador muda a posição dos pólos magnéticos norte e sul do campo da armadura e como o campo produzido pelo enrolamento de campo no estator fica fixo, temos novamente a produção do binário de forças que mantém a mudança dos pólos e consequentemente o movimento do eixo da máquina.

2 – Materiais e Métodos

Os materiais utilizados neste experimento foram:

- Arduino;
- 1 resistor de 3k3Ω e 1 de 220Ω;
- 1 potenciômetro de 1kΩ/5kΩ/10kΩ;
- 1 diodo 1N4004;
- 1 LED;
- 1 Transistor TIP29;
- 1 Motor CC;
- Matriz de contatos (protoboard);
- Multímetro;
- Osciloscópio;
- Fonte de Alimentação 5v.

2.1 – Esquema do circuito

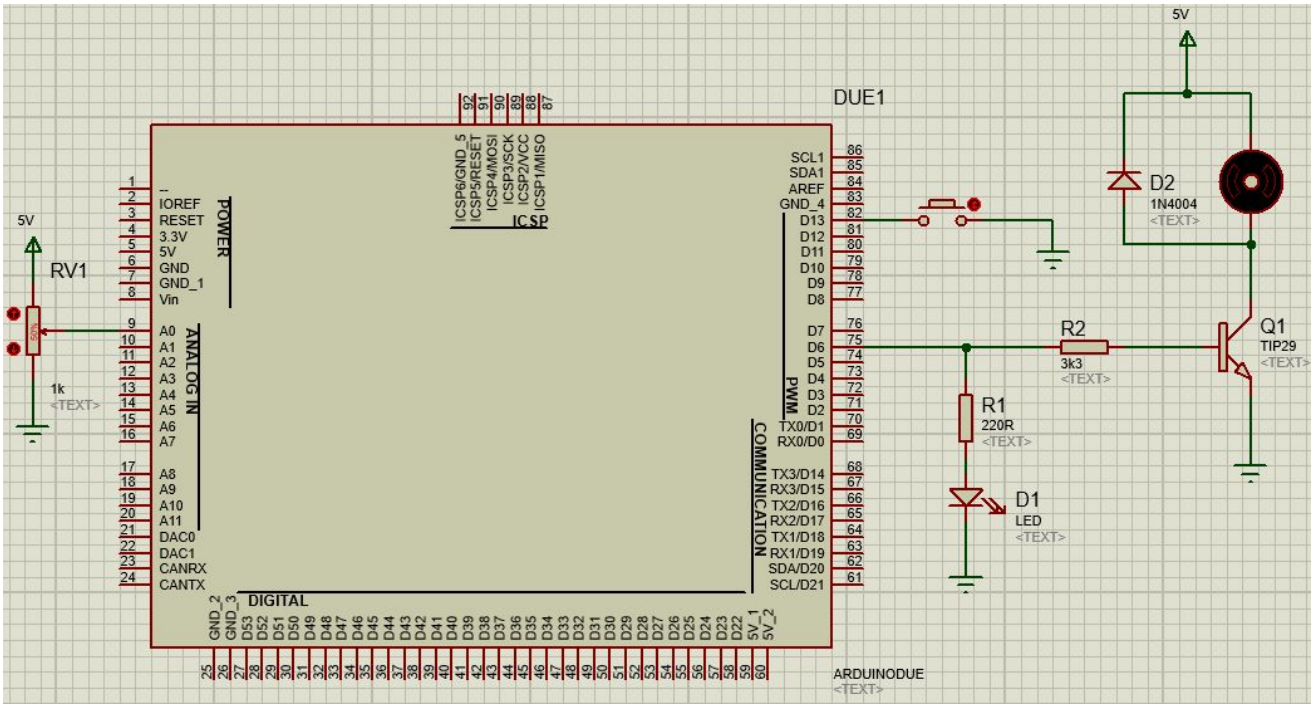

Figura 3: Esquema do circuito montado para o controle do motor CC com PWM.

2.2 – Código

O código escrito para o experimento está descrito no Anexo I (ao final do relatório). As linhas 2-4 fazem a definição dos pinos que são utilizados. nas linhas 6-11 a função *void setup()*, faz a configuração dos pinos de acordo com o regime de trabalho *(INPUT, OUTPUT)* e também a inicialização da porta de comunicação serial do arduino.

No interior da função *void loop()*, a variável *tensao* recebe da função *lerTensao()* o valor de tensão em unidades (inteiros) que vão de 0-330, pois a função *map()* utilizada retorna somente valores inteiros. Posteriormente, a variável *rc* recebe da função *calculaRC()* valores entre 0-255 (0 sempre desligado, 255 sempre ligado), calculados através da função *map()*. Estes valores definem o ciclo de trabalho do pino do motor e dependem diretamente

da tensão (da variável *tensao,* em unidades) lida na porta analógica A0, aplicada pelo potenciômetro. Na linha 34 a tensão é convertida para volts, dividindo o valor da variável *tensao* por 100.

Posteriormente, é chamada a função *exibeMensagem()* responsável pela exibição da mensagem a seguir no monitor serial:

Tensão em A0 = xx.x volts Sinal PWM = xx %

Para garantir que o motor seja ligado somente quando o botão é pressionado, a condição *if-else* avalia através da função *digitalRead()* se o pino ligado ao botão está em estado *high* (1) ou *low (0)*. Como o pino está configurado como *INPUT_PULLUP*, quando o botão estiver pressionado, o estado lógico do pino vai para 0 e então a função *analogWrite()* liga o motor, com a rotação do proporcional ao valor de *rc*. Caso o botão não seja pressionado (aberto), o motor é desligado. O LED ligado ao mesmo pino, permanece ligado com intensidade dependente também da variável *rc.*

3 – Resultados e discussões

Com a variação da tensão aplicada na porta analógica A0 no Arduino, a partir da mudança na chave do potenciômetro, podemos gerar valores em unidade respectivos ao valor de tensão analogica. Estes valores podem ser utilizados como controle do brilho de um LED a partir da geração de sinal PWM. Este sinal é utilizado para ativar uma porcentagem do do total de acordo com o valor da tensão aplicada na porta A0.

4 – Conclusão

Através do experimento, pudemos verificar o funcionamento de um motor de corrente contínua através da geração de sinal PWM pelo arduino. A técnica de modulação por largura de pulso (PWM) é extremamente útil e importante, pois permite a geração de resultados analógicos através de sinais digitais, tendo em vista que o Arduino e outros microcontroladores trabalham com corrente contínua.

5 – Referências

Função analogWrite(). Arduino Language Reference. Acessado: 2 de outubro, 2017. Disponível em <https://www.arduino.cc/en/Reference/AnalogWrite>.

Função analogWriteResolution(). Arduino Language Reference. Acessado: 2 de outubro, 2017. Disponível em <https://www.arduino.cc/en/Reference/AnalogWriteResolution>.

Função analogReadResolution(). Arduino Language Reference. Acessado: 2 de outubro, 2017. Disponível em <https://www.arduino.cc/en/Reference/AnalogReadResolution>.

O que é PWM. Newton Braga. Acessado em 3 de outubro, 2017 Disponível em <http://www.newtoncbraga.com.br/index.php/robotica/5169-mec071a>

ANEXO I - Código para Experimento 3

```
1.  //Definição dos pinos
2.  #define pinAnalogico    A0
3.  #define pinButton       13
4.  #define pinMotor        6 // PWM
5.
6.  void setup() {
7.    //Configuração do regime de trabalho dos pinos e comunicação serial
8.    pinMode(pinButton, INPUT_PULLUP);
9.    pinMode(pinMotor, OUTPUT);
10.   Serial.begin(115200); // Inicia porta serial
11. }
12.
13. float lerTensao() {
14.   //Realiza a leitura retornando valores de unidade entre 0-1023 (0-3.3V)
15.   int valorSensor = analogRead(pinAnalogico);
16.   //Realiza conversão
17.   int  tensao_int = map(valorSensor, 0, 1023, 0, 330); // SOMENTE PARA DUE
18.   float tensao = tensao_int;
19.   return tensao;
20. }
21.
22. void exibeMensagem(float tensao, int rc) {
23.   float rc_perc = rc * 100.0 / 255;
24.   Serial.println("A tensao em AO =  " + (String)tensao + " volts  Sinal PWM = " + (S
    tring)rc_perc + "%");
25. }
26.
27. int calculaRC(float tensao) {
28.   int rc = map(tensao, 0, 330, 0, 255);
29.   return rc;
30. }
31. void loop() {
32.   float tensao = lerTensao();
33.   int rc = calculaRC(tensao);
34.   tensao /= 100.0;
35.   exibeMensagem(tensao, rc);
36.   if (!digitalRead(pinButton)) {
37.     analogWrite(pinMotor, rc);
38.   } else {
39.     analogWrite(pinMotor, 0);
40.   }
41. }
```