SEMANA ACADÊMICA
REVISTA CIENTÍFICA
ISSN 2236-6717

DESENVOLVIMENTO DE UM MEDIDOR DE VAZÃO DIGITAL EMPREGANDO PLACA DE ORIFÍCIO PARA USO EM LINHA DE GASES

[ver artigo online]

Fernando Froes[1]
João Paulo Alves Silva[2]

RESUMO

Inicialmente foi selecionado o sensor de pressão mais adequado para a aplicação. O modelo de sensor escolhido foi o MPX5050DP, este sensor foi conectado a uma placa de orifício. Após isto foi necessário calibrar com o rotâmetro, inicialmente utilizamos um valor genérico para vazão para ver o funcionamento do projeto, após isso foi utilizado o valor correto para que a programação funcionasse corretamente. A programação foi feita para que ao apertar o botão "select", possa ser visualizado a tensão e a vazão ou a tensão e a variação de pressão, sendo assim pode-se ver ambos os dados pelo mesmo dispositivo. No teste final o medidor se comportou bem, estando com valores iguais aos valores do rotâmetro. O projeto foi um sucesso e teve ótimos resultados, sendo um medidor de baixo custo, alta precisão e digital.

Palavras-chave: Arduino. Instrumentação. Medidor de vazão

DEVELOPMENT OF A DIGITAL FLOW METER USING AN ORIFICE PLATE FOR USE IN A GAS LINE

ABSTRACT

Initially the most suitable pressure sensor for an application was selected. The sensor model chosen was the MPX5050DP, this sensor was connected to an orifice plate. After this it was necessary to calibrate with the rotameter, we initially used a generic value for flow to see the operation of the project, after that, the correct value was used by the graph so that the programming worked correctly. The programming was done, so that by pressing the "select" button, the voltage and the flow or the voltage and the pressure variation can be visualized, so you can see both data by the same device. In the final test, the meter behaved well, and with equal values to the values of the rotameter. The project was a success and had great results, being of low cost, high precision and digital.

Keywords: Arduino. Instrumentation. Flowmete

1 Aluno Pesquisador de Doutorado do Departamento de Engenharia de Materiais na Universidade de São Paulo - Escola de Engenharia de Lorena, SP
e-mail: fernando.f@usp.br.
2 Docente na Universidade de São Paulo do Departamento de Engenharia Química na Universidade de São Paulo - Escola de Engenharia de Lorena, SP
e-mail: jpalves80@usp.br.

http://dx.doi.org/10.35265/2236-6717-230-12361

INTRODUÇÃO

Os dispositivos de medição são amplamente utilizados em engenharia. Estes dispositivos medem o quanto de um gás ou líquido passam pela tubulação. São utilizados em muitos processos industriais. Entre eles a placa de orifício (ITUFLUX - INSTRUMENTOS DE MEDIÇÃO, 2012) é um dos medidores de vazão mais utilizados, se destaca por ser um medidor de vazão de baixo custo, baixo índice de manutenção e pode ser aplicado para diversos fluídos. O Arduino também tem um baixo custo e contém vários acessórios de automação com ambos é possível fazer um medidor de vazão de ótima precisão e digital para aplicações empresariais e laboratoriais.

OBJETIVOS

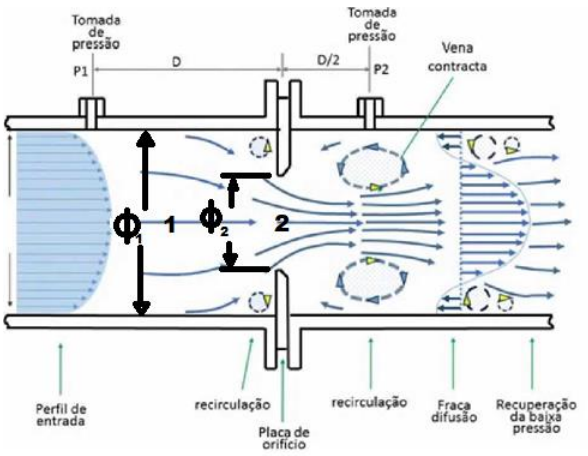
Desenvolver um dispositivo digital de baixo custo para medição de vazão de linhas de gases, baseado em uma placa de orifício integrada ao sensor e microprocessador para o processamento do sinal e indicação de vazão.

REVISÃO BIBLIOGRÁFICA

Placa de orifício

A placa de orifício é um tipo de medidor de vazão por obstrução, do mesmo grupo do medidor de Venturi e bocal. A seguir na Figura 1 tem-se o esquemático de uma placa de orifício.

Figura 1 Placa de orifício com uma corrente de ar interna



Fonte: Adaptado Cortez (2017)

ISSN 2236-6717

Considerando um escoamento em regime permanente e incompreensível em um tubo horizontal. Pode se usar as equações (ÇENGEL; CIMBALA, 2015) de balanço de massa e de Bernoulli entre o local antes da constrição (1) e o local onde a constrição ocorre (2) sendo escritas como:

Balanço de massa:

$$V = A_1V_1 = A_2V_2$$

Equação de Bernoulli ($z_1=z_2$):

$$\frac{P_1}{\rho g} + \frac{V_1^2}{2g} = \frac{P_2}{\rho g} + \frac{V_2^2}{2g}$$

Sendo,

A_1: área do local 1

A_2: área do local 2

V_1: velocidade no local 1

V_2: velocidade do local 2

P_1: pressão do local 1

P_2: pressão do local 2

ρ: densidade do fluído

g: gravidade

Combinando as duas equações podemos isolar a velocidade V_2 e obter:

$$V_2 = \sqrt{\frac{2(P_1 - P_2)}{\rho(1 - \beta^4)}}$$

Que é a velocidade na obstrução sem perda, onde $\beta = \phi_2/\phi_1$ é a razão entre o diâmetro maior (da área 2) e o maior (da área 1). A vazão pode ser determinada simplesmente multiplicando a velocidade pela área descrita pela seguinte equação, onde Vaz é a vazão:

$$Vaz = A_{2*} V_2$$

Com isso vemos que a vazão através de um tubo pode ser determinada por uma restrição ao escoamento e medida pela diminuição da pressão. A queda de pressão entre os dois pontos pode

ser facilmente medida por um transdutor de pressão diferencial. A velocidade obtida em V_2 seria a velocidade máxima possível, visto que a equação não considera que há perdas. A perda pode ser calculada com um fator de correção chamado coeficiente de descarga que é obtido através da calibração com o rotâmetro.

Arduino Uno

O Arduino é uma placa composta por um microcontrolador Atmel, circuitos de entrada/saída e pode ser conectado a um computador por meio de um Ambiente de Desenvolvimento Integrado (IDE) utilizando a linguagem de programação C/C++. Existem diversos modelos de Arduino como: Uno, Mega 2560, Leonardo, Due, ADK, Mega ADK, Nano, Pro Mini e Esplora. A seguir na Tabela 1 tem as especificações de cada modelo:

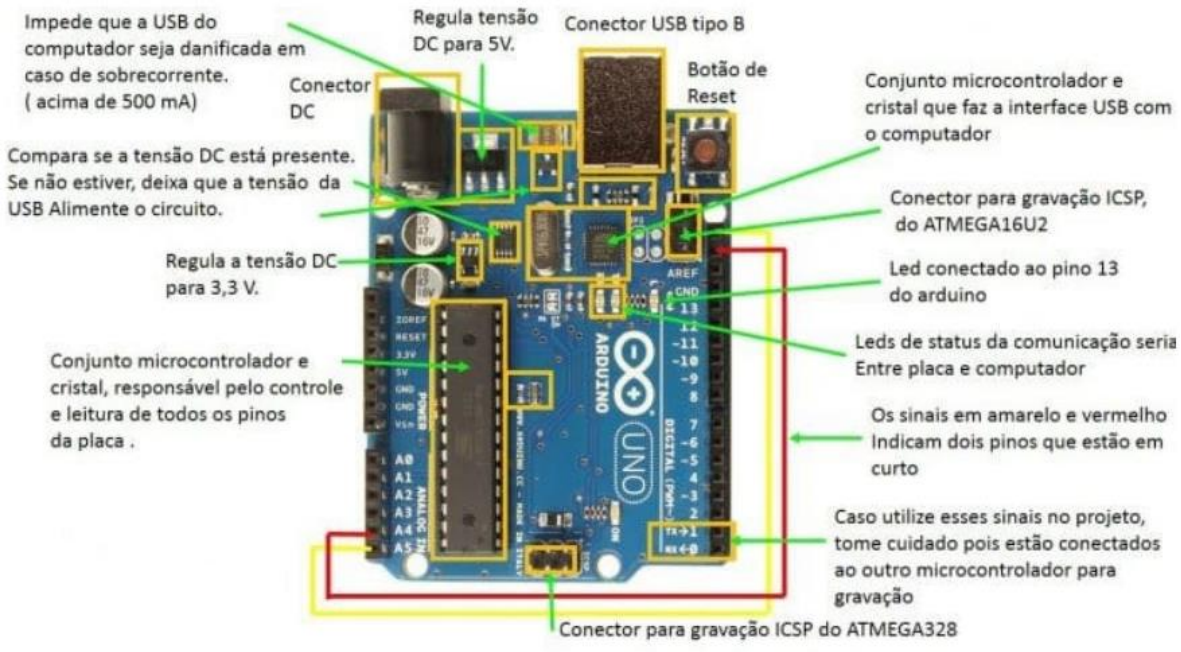
Tabela 1 Especificações de cada modelo Arduino

	UNO	MEGA 2560	LEONARDO	DUE	ADK	NANO	PRO MINI	ESPLORA
Microcontrolador	ATmega328	ATmega2560	ATmega32u4	AT91SAM3X8E	ATmega2560	ATmega168 (versão 2.x) ou ATmega328 (versão3.x)	ATmega168	ATmega32u4
Portas digitais	14	54	20	54	54	14	14	-
Portas PWM	6	15	7	12	15	6	6	-
Portas analógicas	6	16	12	12	16	8	8	-
Memória	32K (0,5K usado pelo bootloader)	256K (8K usado pelo bootloader)	32K (4K usado pelo bootloader)	512K disponível para aplicações	256K (8K usado pelo bootloader)	16K (ATmega168) ou 32K (ATmega328) (bootloader: 2K)	16K (2K usado pelo bootloader)	32K (4K usado pelo bootloader)
Clock	16Mhz	16Mhz	16Mhz	84Mhz	16Mhz	16Mhz	8Mhz (modelo 3.3v) ou 16Mhz (modelo 5v)	16Mhz
Conexão	USB	USB	Micro USB	Micro USB	USB	USB Mini-B	Serial/Módulo USB externo	Micro USB
Conector para alimentação externa	Sim	Sim	Sim	Sim	Sim	Não	Não	Não
Tensão de operação	5V	5V	5V	3.3V	5V	5V	3.3 ou 5V, dependendo do modelo	5V
Corrente máxima portas E/S	40mA	40mA	40mA	130mA	40mA	40mA	40mA	-
Alimentação	7-12Vdc	7-12Vdc	7-12Vdc	7-12Vdc	7-12Vdc	7-12Vdc	3.35-12V (modelo 3.3v) ou 5-12V (modelo 5v)	5V

Fonte: Thomsen (2014)

O modelo do Arduino Uno foi o escolhido para o projeto devido seu baixo custo e suas especificações ideias para o projeto. A seguir na Figura 2 tem-se o Resumo da placa Arduino Uno e suas funções.

Figura 2 Placa Arduino Uno identificada



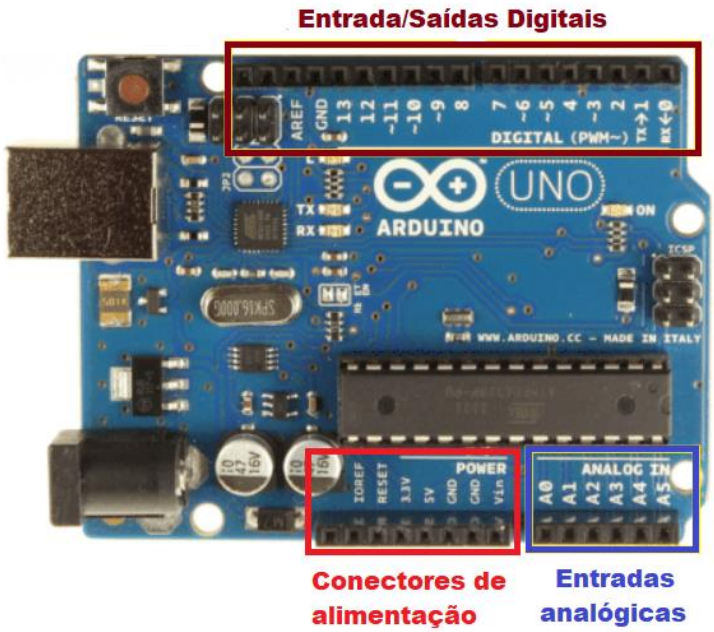
Fonte: Adaptado Souza (2013)

O Arduino Uno (SOUZA, 2013) pode tanto ser ligado por USB ou por uma alimentação externa. A alimentação externa é feita através do conector Jack com positivo no centro e a tensão deve estar entre 7 e 12 V, caso o USB seja alimentado diretamente a tensão deve ser de 5V.

Existe na placa os conectores de alimentação que é composta pelo IOREF, RESET, 3,3 V, 5V, GND e VIN. O conector IOREF fornece uma tensão de referência para que os Shields possam utilizar a interface apropriada, o conector Reset pode ser utilizado para um reset externo da placa Arduino, o conector 3,3 V fornece uma tensão de 3,3 V, o conector 5V fornece uma tensão de 5 V, o conector GND é onde encaixa o pino de referência terra e o VIN é onde a tensão da fonte estará se for alimentado externamente através do conector Jack.

A placa também possui de entrada e saídas digitais, assim como as entradas de entradas e saídas analógicas. A seguir na Figura 3 é exibido a destas portas:

Figura 3 Arduino Uno com os conectores e entradas/saídas analógicas e digitais identificados



Fonte: Adaptado Souza (2013)

LCD Shield

A seguir tem-se na Tabela 2 os pinos utilizados do Arduino e suas funções no LCD Shield:

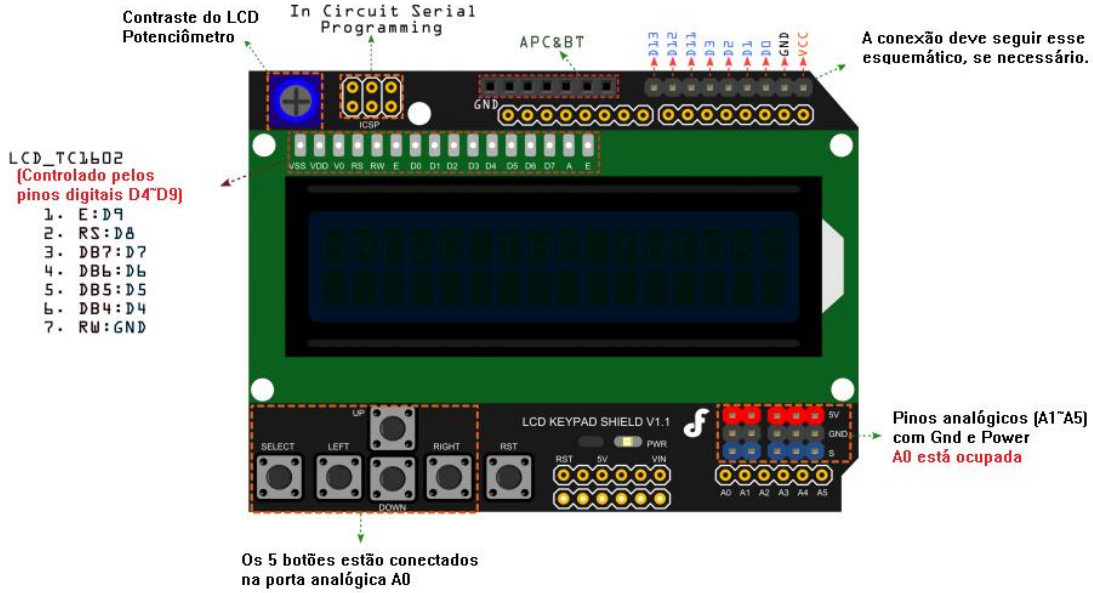
Tabela 2 Os pinos utilizados do arduino e suas respectivas funções no LCD Shield

Pinos	Função
Analog 0	Botões (select, up, right, down e left)
Digital 4	DB4
Digital 5	DB5
Digital 6	DB6
Digital 7	DB7
Digital 8	RS
Digital 9	Enable
Digital 10	Controle da luz de fundo

Fonte: Adaptado de Dfrobot (2017)

As quatro portas lógicas digitais bidirecionais com saída tristate (D4 a D7) (ALBERTINI et al., 2015) são usados como DB4 a DB7 e fazem a transferência/recebimento de dados entre o Arduino e o LCD Shield. A função RS (THOMSEN, 2014) mostra ao display que tipo de dado está transmitindo, em baixo nível (0), os dados são tratados como comando e em alto nível (1) os dados são tratados como caracteres. A função "Enable" é utilizada para determinar o início da transferência de dados entre o Arduino e o LCD Shield. E a porta digital 10 (D10) é usada para controlar a luz de fundo. O pino R/W (Read/Write) é utilizado para determinar se estamos recebendo (lendo) ou transmitindo (escrevendo) dados. A seguir na Figura 4 tem-se o esquemático do LCDShield.

Figura 4 Esquemático do LCDShield



Fonte: Adaptado de Dfrobot (2017)

O potenciômetro é utilizado para calibrar alterar o contraste do display, problemas de visualização geralmente estão ligados ao contraste.

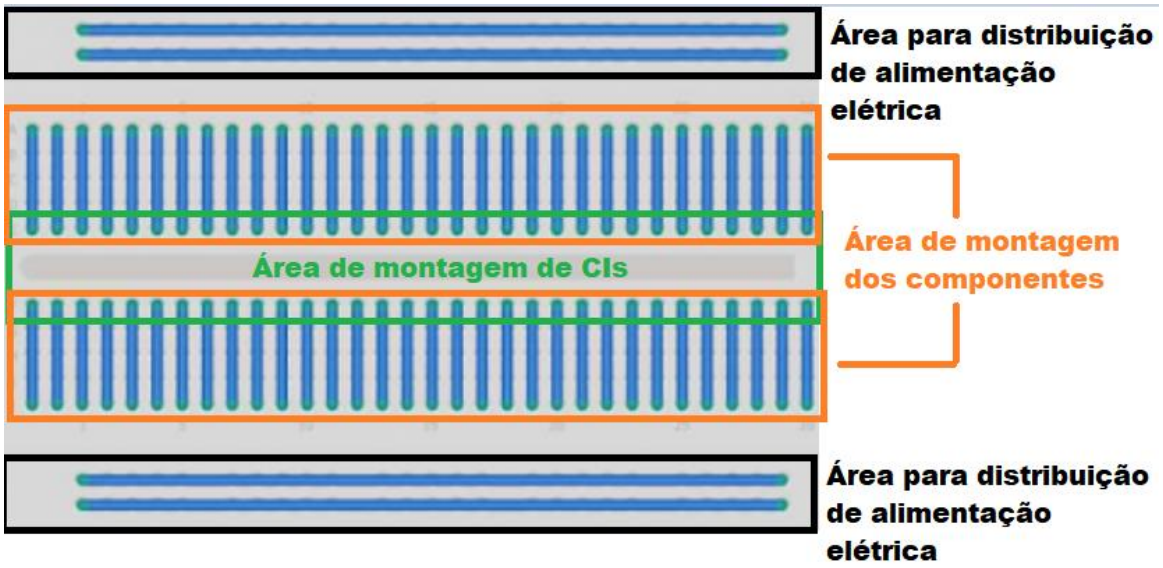
Os 6 pinos ICSP (CAMPOS., 2015) seguem um padrão fixo, sendo 3 pintos de tensão e controle (VCC, GND e RESET) e três pinos de conexão ponto a ponto (MOSI, MISO e SCK)

por onde há transferência de dados e pulso/clock de conexão. O nome ICSP ou ISP referente à capacidade de transferir programas para um componente enquanto ele está instalado em um sistema através de protocolos seriais.

Protoboard

A protoboard (ROBOCORE TECNOLOGIA LTDA, 2017) possibilita montar vários circuitos sem a necessidade de soldar os componentes. É muito importante para quem está iniciando um projeto e precisa fazer modificações. O seu funcionamento se baseia nas ligações dos pontos indicado nas linhas azuis, da Figura 5 tem-se as três áreas disponíveis na protoboard, a área de alimentação elétrica (as duas linhas superiores e inferiores), a área de montagem dos componentes (colunas) e a área de montagens de Circuitos Integrados (parte central).

Figura 5 Protoboard separada por áreas



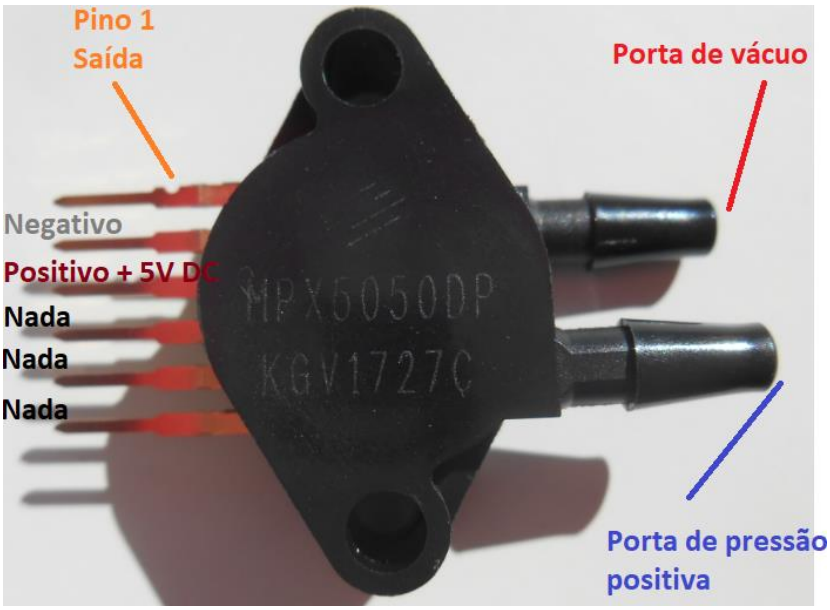
Fonte: Adaptado Robocore Tecnologia Ltda (2017)

Sensor diferencial de pressão MPX5050DP

O MPX5050DP mede uma variação de pressão de 0 a 50 kPa que é reproduzida como um sinal que varia de 0,2 a 4,7 V; necessitando de uma alimentação de 5 V (FREESCALE SEMICONDUCTOR, 2010). Abaixo na Figura 6 tem a foto do sensor com as funções das portas

e pinos indicados, na saída (pino 1) os valores vão de 0,2 a 4,7 sendo diretamente proporcionais a tomada de pressão (0 a 50kPa), no negativo (pino 2) é onde o dispositivo é aterrado, no positivo (pino 3) ele é ligado a uma tensão de 5 V e nos demais pinos (4, 5 e 6) não tem função de acordo com manual do fornecedor.

Figura 6 Foto do sensor MPX5050DP com as indicações das portas e pinos



Fonte: autoria própria

MATERIAIS E MÉTODOS

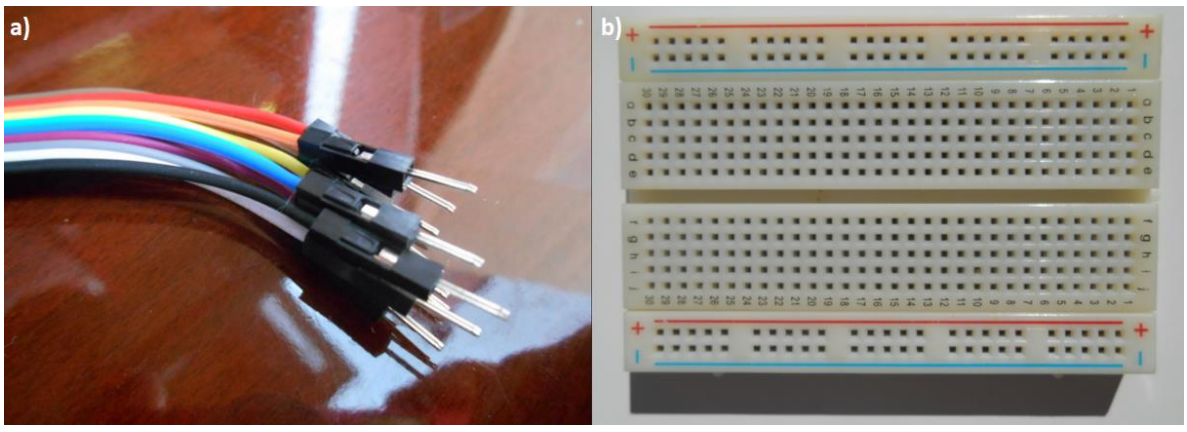
Inicialmente foi selecionado o sensor de pressão mais adequado para a aplicação. O modelo de sensor escolhido foi o MPX5050DP, este sensor foi conectado a uma placa de orifício. A seguir na Figura 7 tem-se a imagem da placa de orifício utilizada.

Figura 7 Placa de orifício utilizada para medir a vazão

Fonte: autoria própria

Com isso foi necessário ligar o sensor ao Arduino através da protoboard de 400 pontos e jumpers macho-macho para conexão. A seguir na Figura 8 tem a foto dos jumpers (a) e da protoboard (b):

Figura 8 Foto do jumper (a) e da protoboard (b)



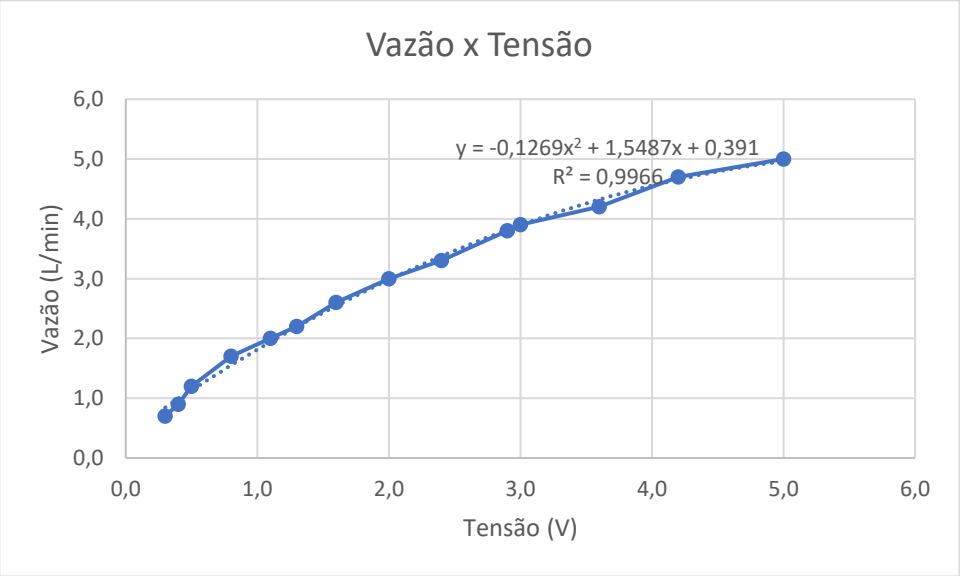
Fonte: autoria própria

O Arduino por sua vez foi ligado ao LCD Shield com tecla para Arduino tendo um perfeito encaixe em cima do Arduino Uno. O Shield é montado diretamente sobre a placa e as 6 teclas utilizam a entrada analógica A0 e deixa livre as portas digitais e analógicas.

Após a montagem foi necessário fazer a programação no software da Arduino. No apêndice segue a programação (DAMAS, 2007 e ARDUINO 2018) utilizada na plataforma IDE para o funcionamento do projeto (esta programação já foi calibrada com o rotâmetro).

Após isto foi necessário calibrar com o rotâmetro, inicialmente utilizamos um valor genérico para vazão para ver o funcionamento do projeto, após isso foi utilizado o valor correto pelo gráfico para que a programação funcionasse corretamente, o gráfico a seguir na Figura 9 mostra a relação entre a tensão e a vazão:

Figura 9 Gráfico Vazão x Tensão

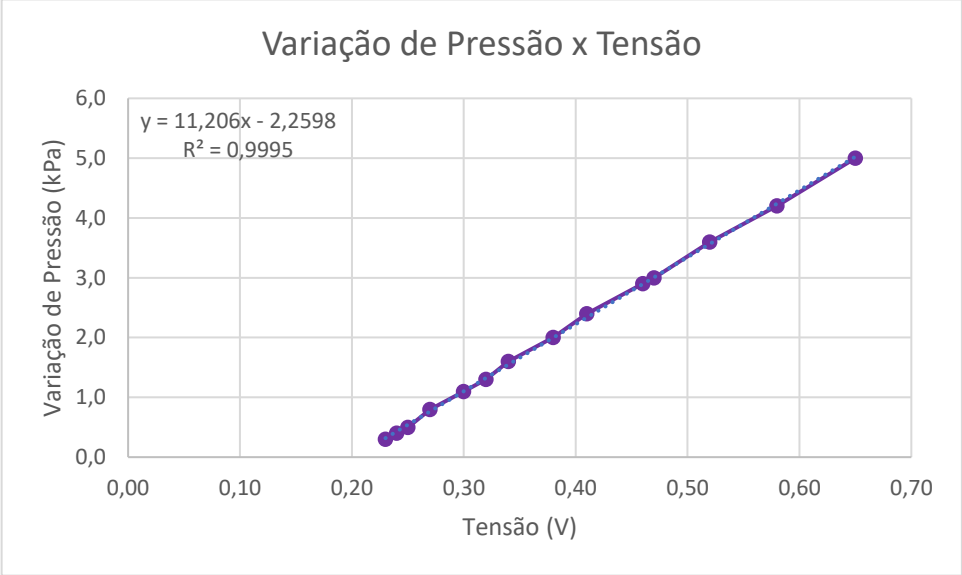


Fonte: autoria própria

CONSIDERAÇÕES FINAIS

Normalmente na literatura a calibração é feita medindo a variação de pressão, neste caso utilizamos a variação de tensão pois tem o mesmo significado. Por ser um medidor digital a pressão está diretamente ligada a tensão, abaixo tem-se o gráfico da tensão pela variação de pressão. Nota-se que o resultado é linear com uma alta precisão:

Figura 10 Gráfico da relação entre a variação de pressão e a tensão



Fonte: autoria própria

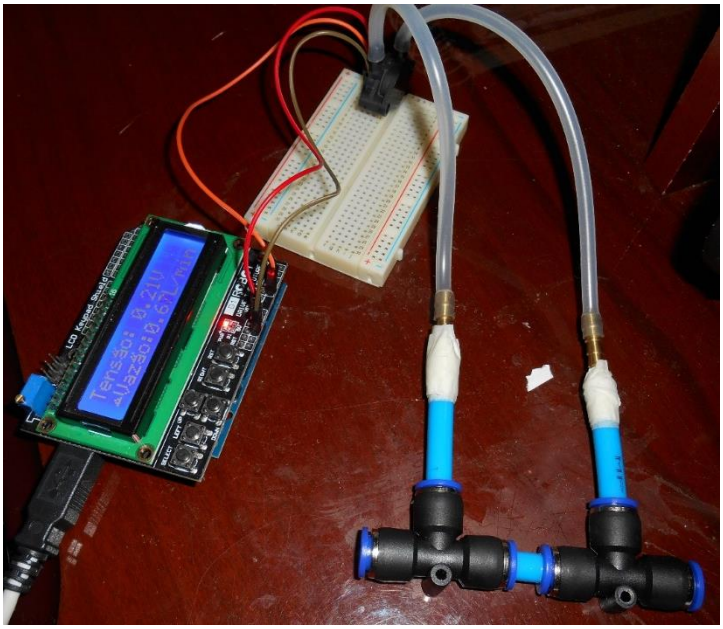
Também é notável que a equação que descreve a calibração com o rotâmetro tem um coeficiente de determinação próximo de 1, o que significa que nossa medição digital será precisa.

A programação foi feita para que ao apertar o botão “select”, possa ser visualizado a tensão e a vazão ou a tensão e a variação de pressão, sendo assim pode-se ver ambos os dados pelo mesmo dispositivo, os detalhes delas estão escritos as frente em linhas de comentários.

A seguir na Figura 11 tem a foto do projeto completo:

ISSN 2236-6717

Figura 11 Projeto (Arduino + LCD Shield + jumper + protoboard + placa de orifício)



Fonte: Autoria própria

No teste final o medidor se comportou bem, estando com valores iguais aos valores do rotâmetro. O projeto foi um sucesso e teve ótimos resultados, sendo um medidor de baixo custo, alta precisão e digital.

Sugere-se em trabalhos futuros utilizar as portas extras, o motor de passo e fazer um controlador de vazão de gás automático que fixe a vazão inserida no display e mantenha constante.

REFERÊNCIAS BIBLIOGRÁFICAS

ALBERTINI, Bruno de Carvalho et al. **Buffer Tri-State**. 2015. Disponível em: <http://eaulas.usp.br/portal/video.action?idItem=7421>. Acesso em: 23 jul. 2018.

ARDUINO. **Documentação de Referência da Linguagem Arduino.** 2018. Disponível em: <https://www.arduino.cc/reference/pt/>. Acesso em: 26 jul. 2018.

CAMPOS., Augusto. **Entendendo os 6 pinos de ICSP dos Arduinos.** 2015. Disponível em: <https://br-arduino.org/2015/05/arduino-icsp-attiny-atmega.html>. Acesso em: 23 jul. 2018.

ÇENGEL, Yunus A.; CIMBALA, John M.. **Mecânica dos fluidos**: Fundamentos e aplicações. 3. ed. Porto Alegre: Amgh, 2015. (9788580554908). Tradução: Fábio Saltara, Jorge Luis Baliño, Karl Petter Burr..

CORTEZ, Gilberto Garcia. **Medidas de vazão em líquidos mediante o uso da placa de Orifício, Venturi e Rotâmetro**: Lorena: Escola de Engenharia de Lorena - Usp, 2017. 27 slides, color.

DAMAS, Luís. **Linguagem C.** Ltc, 2007.

DFROBOT (Shanghai). **LCD KeyPad Shield For Arduino SKU: DFR0009.** 2017. Disponível em: <https://www.dfrobot.com/wiki/index.php/LCD_KeyPad_Shield_For_Arduino_SKU:_DFR0009>. Acesso em: 23 jul. 2018.

FREESCALE SEMICONDUCTOR (Campinas). **Integrated Silicon Pressure Sensor On-Chip Signal Conditioned, Temperature Compensated and Calibrated.** 2010. Disponível em: <www.freescale.com>. Acesso em: 23 jul. 2018.

ITUFLUX - INSTRUMENTOS DE MEDIÇÃO (Itu). **Placa de Orifício.** 2012. Disponível em: <http://www.ituflux.com.br/produto/placa-de-orificio-1>. Acesso em: 25 jul. 2018.

ROBOCORE TECNOLOGIA LTDA (São Caetano do Sul). **Como utilizar uma Protoboard.** 2017. Disponível em: <https://www.robocore.net/tutoriais/como-utilizar-uma-protoboard.html>. Acesso em: 23 jul. 2018.

SOUZA, Fábio. **Arduino Uno.** 2013. Disponível em: <https://www.embarcados.com.br/arduino-uno/>. Acesso em: 23 jul. 2018.

THOMSEN, Adilson. **Ligando Display LCD 16×2 ao PIC 16F628A.** 2014. Disponível em: <https://www.filipeflop.com/blog/display-lcd-16x2-pic-16f628a/>. Acesso em: 23 jul. 2018.

THOMSEN, Adilson. **Qual Arduino Comprar? Conheça os Tipos de Arduino.** 2014. Disponível em: <https://www.filipeflop.com/blog/tipos-de-arduino-qual-comprar/>. Acesso em: 23 jul. 2018.

ISSN 2236-6717

APÊNDICE A: Programação

```
#include <LiquidCrystal.h> //inclui a biblioteca do LCD
byte delta[8] = // criação da função delta no display para indicar variação. Os locais onde aparece 0 não acendem e onde aparece 1 acende, criando assim o caractere delta.
{
 B00000,
 B00000,
 B00100,
 B01010,
 B11111,
 B00000,
 B00000,
 B00000

};

byte atio[8] = //criação do caracetere "ã" para acentuar
{
 B00101,
 B01010,
 B11110,
 B00001,
 B01111,
 B10001,
 B01111,
 B00000

};
```

ISSN 2236-6717


```
// inicia a  biblioteca associando aos pinos necessários para o funcionamento do Display LCD Shield
// seleciona os pinos do arduinos que serão conectados
const int rs = 8, en = 9, d4 = 4, d5 = 5, d6 = 6, d7 = 7; //pinos
LiquidCrystal lcd(rs, en, d4, d5, d6, d7); //em um display convencional são usados os pinos 8, 9, 4, 5, 6, 7 no lugar dos 8, 9, 4, 5, 6, 7 respectivamente.
int i = 1;

void setup() {
 // inicia a comunicação serial a 9600 bits por segundo
 Serial.begin(9600);

 lcd.createChar(0, atio); // cria o caractere ã e associa ao 0
 lcd.createChar(1, delta); //cria o caractere delta e associa ao 1
 lcd.begin(16, 2); // Inicializa do Display LCD de 2 linhas e 16 colunas
 lcd.print("Tensao: "); //escreve a palavra Tensao no display
 lcd.setCursor(4, 0); //posiciona o cursor na posição 4, 0 (coluna 4,linha 0) no LCD
 lcd.write(byte(0)); //escreve o caractere ã
 lcd.setCursor(0, 1);  //posiciona o cursor na posição 0, 1 (coluna, linha)
 lcd.write(byte(1)); //escreve o caractere delta
 lcd.setCursor(1, 1); //posiciona o cursor na posição 1, 1 (coluna, linha)
 lcd.print("Pressao: "); //escreve a palavra Pressao no display LCD
 lcd.setCursor(6, 1); //posiciona o cursor na posição 6, 1 (coluna, linha)
 lcd.write(byte(0)); //escreve o caractere ã

  }
void loop() {

 int valordatecla = analogRead (A0); // Lê o pino A0:
 float sensorValue = analogRead(A1);  // Lê o pino A1:
```

float tensao = sensorValue*0.0049; // Cada unidade do "sensorValue" (0-1024) representa 49 mV, ou seja, deve-se usar essa conversão para obter o valor em Volts

float pressao = (tensao-0.20)*11.11; // cada valor de tensao equivale a aproximadamente 11,11 Kpa

float vazao = -0.1269*tensao*tensao+1.5487*tensao+0.361; //equação da placa de orificio que associa a tensão ao valor da vazão em L/min podendo ser ajustada para cada placa em particular.

Serial.print("tensão: "); //escreve tensão na tela do computador (Ferramentas>Monitor serial).

Serial.print(tensao); // mostra o valor de leitura na tela do computador da tensão (Ferramentas>Monitor serial) PS: foi retirado o valor da pressão atmosférica

Serial.println(" V"); //mostra a unidade da tensão e pula para próxima linha

Serial.print("pressão: ");

Serial.print(pressao); // mostra o valor de leitura na tela do computador da pressão

Serial.println(" kPa"); //mostra a unidade da pressão e pula para próxima linha

Serial.print("vazão : ");//escreve vazão na tela do computador

Serial.print(vazao); // mostra o valor de leitura na tela do computador da vazão

Serial.println(" L/min"); //mostr ao valor da leitura de vazão na tela do computador

Serial.println ("\t"); //Escreve uma tabulação e inicia a escrita na próxima linha

while (valordatecla>506 && valordatecla<742){ //enquanto o valor estiver entre 506 e 742...

i=i+1; //soma +1 na variável i

delay (100); //tempo a cada verificação

break; //para o laço

}

if(i % 2 == 0) { //se i for par...

lcd.setCursor(8, 0); //posiciona o cursor na posição 8, 0 (coluna, linha)

lcd.print(tensao, 2); //mostra o valor da tensão no display

ISSN 2236-6717


```
lcd.setCursor(12, 0); //posiciona o cursor na posição 12, 0 (coluna, linha)
lcd.print ("V"); //escreve V no display
lcd.setCursor(1, 1); //posiciona o cursor na posição 1, 1 (coluna, linha)
lcd.print("Pressao: "); //escreve a palavra Pressao no display LCD
lcd.setCursor(6, 1); //posiciona o cursor na posição 6, 1 (coluna, linha)
lcd.write(byte(0)); //escreve o caractere ã
lcd.setCursor(10, 1); //posiciona o cursor na posição 10, 1 (coluna, linha)
lcd.print(pressao, 1); //mostra o valor da pressão no display (foi retirado o valor da pressão atmosférica) em kPa
lcd.setCursor(13, 1); //posiciona o cursor na posição 13, 1 (coluna, linha)
lcd.print ("kPa"); //escreve kPa no display

 delay(100);       // tempo a cada leitura em ms
 }

else {//senão...
lcd.setCursor(8, 0); //posiciona o cursor na posição 8, 0 (coluna, linha)
lcd.print(tensao, 2); //mostra o valor da tensão no display
lcd.setCursor(12, 0); //posiciona o cursor na posição 12, 0 (coluna, linha)
lcd.print ("V"); //escreve V no display
lcd.setCursor(1, 1); //posiciona o cursor na posição 1, 1 (coluna, linha)
lcd.print("Vazao: "); //escreve a palavra Vazao no display LCD
lcd.setCursor(4, 1); //posiciona o cursor na posição 4, 1 (coluna, linha)
lcd.write(byte(0)); //escreve o caractere ã
lcd.setCursor(7, 1); //posiciona o cursor na posição 7, 1 (coluna, linha)
lcd.print(vazao, 2); //mostra o valor da vazão no display com duas casas decimais de precisão
lcd.setCursor(11, 1); //posiciona o cursor na posição 11, 1 (coluna, linha)
lcd.print ("L/min"); //escreve L/min no display

delay(100);       // tempo a cada leitura em ms
```

ISSN 2236-6717


```
    }
}
```