Vitor Amadeu Souza

Enviando a medição de

Sensores

para monitoramento no

VC#

Com a Raspberry Pi programada em Python

© 2021 by Cerne Tecnologia e Treinamento Ltda.

© 2021 by Vitor Amadeu Souza

Nenhuma parte desta publicação poderá ser reproduzida sem autorização prévia e escrita de **Cerne Tecnologia e Treinamento Ltda.** Este livro publica nomes comerciais e marcas registradas de produtos pertencentes a diversas companhias. O editor utiliza as marcas somente para fins editoriais e em benefício dos proprietários das marcas, sem nenhuma intenção de atingir seus direitos.

Setembro de 2021

Direitos reservados por:
Cerne Tecnologia e Treinamento Ltda
Produção: *Cerne Tecnologia e Treinamento*
E-mail da Empresa: *cerne@cerne-tec.com.br*
Home Page: *www.cerne-tec.com.br.com.br*
Atendimento ao Consumidor: *sac@cerne-tec.com.br*
Contato com o Autor: *vitor@cerne-tec.com.br*



FEITO NO BRASIL

"Envia a tua luz e a tua verdade, para que me guiem; levem-me elas ao teu santo monte, e à tua habitação."

Sl 43:3

Kits Didáticos e Gravadores da Cerne Tecnologia

A Cerne tecnologia têm uma linha completa de aprendizado para os microcontroladores da família PIC, 8051, Holtek, dsPIC, ARM, Arduino, etc. Veja os detalhes de um de nossos kits.



Kit Cerne Arduino

- Microcontrolador ATMEGA8;
- Comunicação serial RS232;
- Alimentação de 12V;
- Pinos de I/O;
- Gravação ICSP.

Uma linha completa de componentes para o desenvolvimento de seus projetos eletrônicos como displays, PICs, botões, leds, cristais e etc. Visite a nossa página na Internet, no endereço **www.cerne-tec.com.br** e conheça melhor nossos serviços e produtos.

Cerne
Conhecimento para o Desenvolvimento

www.cerne-tec.com.br

Sumário

I. Metodologia de desenvolvimento **7**

1. Introdução 7

II. Comunicação serial **8**

1. Módulo conversor TTL<→USB CH340 8

2. Exemplo de transmissão serial 9

III. Comunicando com o ADC externo **10**

1. Conceito 10

2. Esquema elétrico 16

3. Fluxograma 17

4. Código fonte 18

IV. Amperímetro **25**

V. Barômetro **39**

VI. Medindo a concentração de Benzeno **50**

VII. Célula de carga **66**

VIII. Medição de dióxido de Carbono **81**

IX. Medição de Monóxido de Carbono **100**

X. Decibelímetro **119**

XI. Dinamômetro **137**

XII. Medindo Etanol em ppm **147**

XIII. Medindo o fluxo de massa **161**

XIV. Gaussímetro **176**

XV. Medindo a concentração de GLP **192**

XVI. Medindo a concentração de H_2 **208**

XVII. Higrômetro **224**

XVIII. Medindo a concentração de Ozônio **235**
XIX. Medição de pH **251**
XX. Termômetro **266**
XXI. Medindo a radiação ultravioleta **277**
XXII. Voltímetro **294**
XXIII. Monitoramento no VC# **306**

Capítulo I

Metodologia de desenvolvimento

1. Introdução

A proposta deste livro é enviar mensagens via comunicação serial para um programa de monitoramento desenvolvido no Visual C# (VC#) com a medição de diversos sensores listados no sumário da obra usando para isso um conversor TTL<->USB com a programação na Raspberry Pi baseada no Python.

Recomenda-se a leitura prévia ou conhecimento dos assuntos abordados na referência *Aplicações eletrônicas na Raspberry Pi Zero - Programado em Python (2017)* do mesmo autor e editora.

Capítulo II

Comunicação serial

1. Módulo conversor TTL<→USB CH340

Para possibilitar a comunicação entre a Raspberry Pi e o computador no qual o programa de monitoramento será desenvolvido faz-se necessário o uso do módulo conversor a seguir, baseado no circuito integrado CH340 (é possível usar outros modelos compatíveis).



Conecte o pino RX do módulo CH340 ao pino TX da Raspberry Pi (localize através do silk da placa) assim como o pino de GND.

2. Exemplo de transmissão serial

O exemplo a seguir possibilita o envio de uma string via canal serial de comunicação. Note que para isso é necessário importar a biblioteca serial logo no início do programa. Neste exemplo o baud rate (taxa de comunicação) utilizado foi de 9600 bps.

```
import serial
com = serial.Serial("/dev/ttyAMA0", 9600)
com.write("Teste de comunicação serial\n\r")
com.write(26)
```

Para observar a string enviada pelo Arduino sendo recebida pelo PC abra o Arduino IDE e em seguida o Monitor Serial indo no menu Ferramentas.

Capítulo III

Comunicando com o ADC externo

1. Conceito

A série de ADCs (Conversor Analógico para Digital) da Microchip é segmentada por aplicação, havendo modelos específicos de acordo com a necessidade do projeto. Mais informações podem ser obtidas visitando o site do fabricante, no endereço www.microchip.com seção *Products -> Data Converters -> Analog-to-Digital Converters*.

Nesta literatura, foi utilizado um modelo da linha de conversores do tipo Delta-sigma que apresentam como principal característica o fato de possuírem uma alta resolução, mas o número de amostras por segundo (samples/sec) é relativamente baixo, estando na faixa de 60 samples/sec.

O modelo adotado foi o MPC3551 com uma entrada analógica diferencial e 21 bits de resolução além de 1 bit de sinal. A pinagem deste modelo está disponível na figura abaixo.

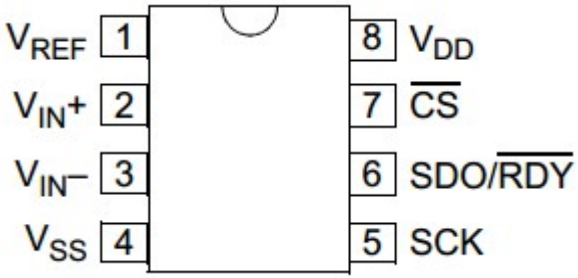


Figura 2.1: Pinagem do MCP3551
Fonte: MCP3551 Datasheet

A alimentação é feita através dos pinos VDD e VSS, podendo variar na faixa de 2,7V a 5,5V. A comunicação SPI com o Raspberry é estabelecida através dos pinos CS, SCK e SDO. A entrada analógica diferencial é fornecida através dos pinos V_{IN+} e V_{IN-}. O pino V_{REF} determina a tensão máxima de referência para o conversor ADC, onde este não deve ser superior a tensão aplicada em VDD.

Obs: *Caso o modo diferencial não seja utilizado, a entrada* V_{IN-} *pode ser conectada ao GND. Neste caso, a entrada* V_{IN+} *poderá variar de GND a* V_{REF}.

A próxima figura mostra o diagrama de blocos do MCP3551.

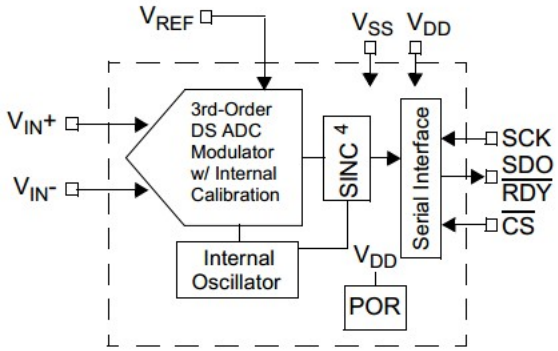


Figura 2.2: Diagrama de blocos interno
Fonte: MCP3551 Datasheet

Considerando que V_{REF}=5V a conversão da entrada analógica para a palavra digital em 21 bits é dada de acordo com o gráfico abaixo.

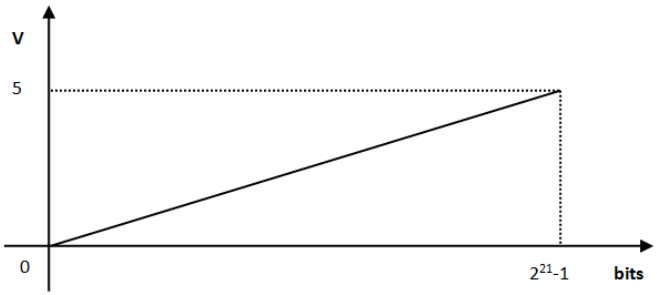


Figura 2.3: Gráfico que relaciona a conversão em bits de acordo com a tensão de entrada

Este gráfico é válido para V_{IN+}>V_{IN-}. Caso contrário, o gráfico tem o mesmo comportamento, no entanto, o bit 22 da conversão fica em

1 indicando esta condição. A tabela a seguir procura esclarecer este funcionamento. Observe que além dos 22 bits da conversão ADC, há os bits OVL e OVH. A função destes bits é indicar uma condição de overflow (transbordo) ficando em nível alto OVL caso a tensão V_{IN} seja menor que $-V_{REF}$ e OVH em nível alto caso seja maior que V_{REF}. Fora destas condições, estes bits ficam em nível 0. Estes bits assim como o resultado da conversão são lidos através do barramento SPI.

Analog Input Voltage	OVL	OVH	Digital Output Code																						Decimal Code	Hexa
	B23	B22	B21	B20	B19	B18	B17	B16	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0		*B[23:0]*
Vref +1 LSB	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2097153	600001
Vref	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2097152	600000
Vref - 1 LSB	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2097151	1FFFFF
2 LSB	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	000002
1 LSB	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	000001
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	000000
-1 LSB	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1	3FFFFF
-2 LSB	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	-2	3FFFFE
-Vref	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-2097152	200000
-Vref -1LSB	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-2097153	9FFFFF
-Vref -2LSB	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	-2097154	9FFFFE

Tabela 2.1: Tabela de conversão do ADC de acordo com a tensão de entrada
Fonte: MCP3551 Datasheet

Observe que o gráfico da figura 2.3 trata-se de uma função linear, definida segundo a função a seguir.

$$ADC = \frac{(2^{21} - 1).V_{IN}}{V_{REF}}$$

Figura 2.4: Função de conversão V_{IN} para bits do ADC
Fonte: MCP3551 Datasheet

Em que ADC é a palavra de conversão do ADC em 21 bits. O campo V_{REF} é a tensão de referência que pode estar na faixa entre VDD e VSS e V_{IN} é a diferença entre a V_{IN+} e V_{IN-}.

Acompanhe os exemplos abaixo que mostram os resultados de conversão observados em 21 bits de acordo com a tensão de entrada V_{IN} (V_{IN+} - V_{IN-}) e a tensão de referência V_{REF}.

1) Determine o resultado da conversão ADC sabendo-se que V_{IN}=1V e V_{REF}=3,3V.

$$ADC = \frac{(2^{21} - 1).V_{IN}}{V_{REF}}$$

$$ADC = \frac{(2^{21} - 1).1}{3,3}$$

$$ADC = 635500_{10} = 0x9B26C$$

Obs: *O resultado acima é o mesmo, está apresentado apenas na base 10 e 16.*

Obs: *No exemplo proposto,* V_{REF}=3,3V e V_{IN-}=GND.

2) Determine o resultado da conversão ADC sabendo-se que V_{IN}=0,5V e V_{REF}=3V.

$$ADC = \frac{(2^{21} - 1).V_{IN}}{V_{REF}}$$

$$ADC = \frac{(2^{21} - 1).0,5}{3}$$

$$ADC = 349525_{10} = 0x55555$$

A comunicação SPI feita com o MCP3551 permite apenas operação de leitura, já que o pino de entrada não está disponível. A próxima figura mostra uma carta de tempos com uma comunicação realizada com o MCP3551.

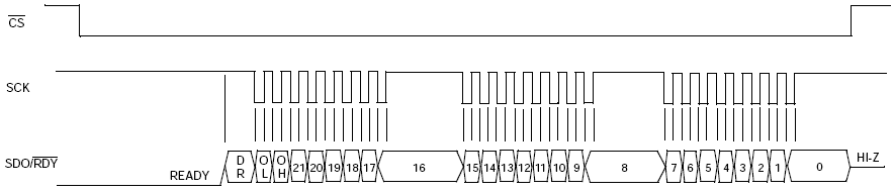


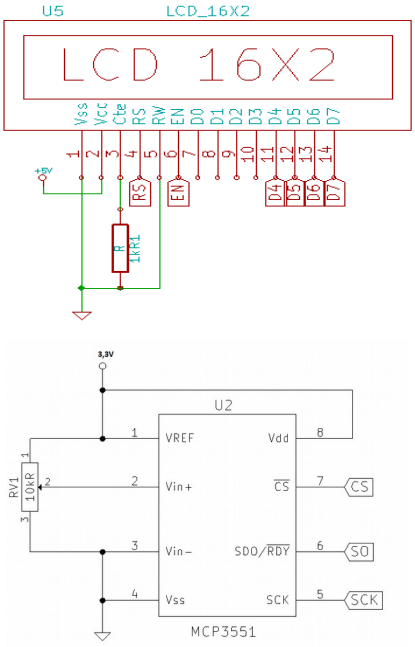
Figura 2.5: Carta de tempos da comunicação com o MCP3551
Fonte: MCP3551 Datasheet

Observe que o primeiro passo consiste em deixar a linha CS em nível baixo, de modo a selecionar o MCP3151 durante todo o intervalo da comunicação. Em seguida, deve-se aguardar SDO/RDY ir para nível baixo, o que indica que o circuito integrado está pronto a fornecer o resultado da conversão ADC. Finalmente são gerados 24 pulsos de clock para ler o conteúdo da conversão do ADC, onde a leitura deverá ser feita no instante que a linha de clock estiver em nível baixo.

2. Esquema elétrico

O esquema elétrico proposto para este experimento pode ser verificado na próxima figura, no qual um LCD é utilizado como mostrador e o ADC MCP3551 converte a tensão analógica proveniente de um trimpot para o formato digital. Tal trimpot pode fornecer uma tensão entre GND e V_{REF}.

Outro sensor pode ser conectado ao ADC, bastando remover o trimpot e ligá-lo a entrada analógica desde que a tensão esteja na faixa de tensão definida em V_{REF}.



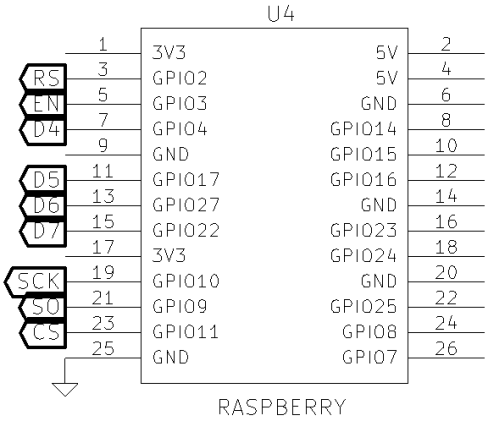
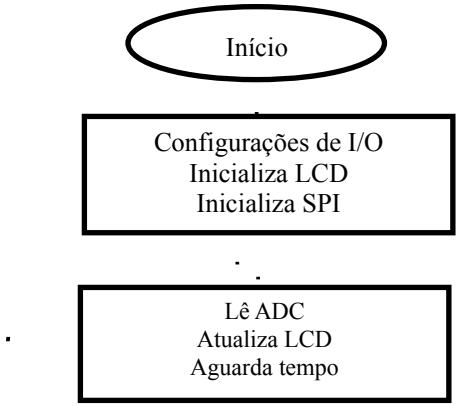


Figura 2.6: Esquema elétrico proposto

3. Fluxograma

O exemplo proposto comunica periodicamente com o MCP3551 para em seguida mostrar o valor da conversão no LCD. O algoritmo proposto está apresentado no fluxograma a seguir.



4. Código fonte

Baseado no esquema elétrico e no fluxograma apresentado está disponibilizado a seguir o código fonte deste experimento.

```
#*  *  *  *  *  *  *  *  *  *  *  *  *
#*          Cerne Tecnologia           *
#*        www.cerne-tec.com.br         *
#*      Projetos em Raspberry - ADC    *
#*  *  *  *  *  *  *  *  *  *  *  *  *

#*  *  *  *  *  *  *  *  *  *  *  *  *
#*         Declaração de I/Os          *
#*  *  *  *  *  *  *  *  *  *  *  *  *
RS=3
EN=5
D4=7
D5=11
D6=13
D7=15
SCK=19
SO=21
CS=23

#*  *  *  *  *  *  *  *  *  *  *  *  *
#*       Função de inicialização       *
#*  *  *  *  *  *  *  *  *  *  *  *  *
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)
spi_inic()
lcd_inic()

#*  *  *  *  *  *  *  *  *  *  *  *  *
#*           Função de loop            *
#*  *  *  *  *  *  *  *  *  *  *  *  *
while true:
  adc=spi_read_adc()
```

```
  if(adc & 0x00800000):      #Testa bit 23. OVL?
    lcd_texto(1,1,"Overflow!       ")
    lcd_texto(2,1,"Vin<Vref        ")
    time.sleep(1)

  else if(adc & 0x00400000): #Testa bit 22. OVH?
    lcd_texto(1,1,"Overflow!       ")
    lcd_texto(2,1,"Vin>Vref        ")
    time.sleep(1)

  else:                      #Condição normal
    lcd_texto(1,1,"%s",adc)
    lcd_texto(2,1,"-Vref<Vin<Vref ")
    time.sleep(1)

#*  *  *  *  *  *  *  *  *  *  *  *  *
#*       Inicialização da SPI        *
#*  *  *  *  *  *  *  *  *  *  *  *  *
def spi_inic():
  GPIO.setup(SCK,GPIO.OUT)
  GPIO.setup(SO,GPIO.IN,pull_up_down=GPIO.PUD_UP)
  GPIO.setup(CS,GPIO.OUT)
  GPIO.output(SCK,1)
  GPIO.output(SI,1)
  GPIO.output(CS,1)

#*  *  *  *  *  *  *  *  *  *  *  *  *
#*  Rotina de envio de byte pela SPI *
#*  *  *  *  *  *  *  *  *  *  *  *  *
def spi_write_byte(byte_):
   contador=0b10000000
   while(contador):
      GPIO.output(SI,byte_ & contador)
      GPIO.output(SCK,1)
      GPIO.output(SCK,0)
      contador=contador>>1
```

```
#*  *  *  *  *  *  *  *  *  *  *  *  *
#*     Rotina de leitura do ADC      *
#*  *  *  *  *  *  *  *  *  *  *  *  *
def spi_read_adc():
   aux=0x00800000
   conv=0

   GPIO.output(CS,0)

   while GPIO.input(SO):
       time.sleep(0.010)

   for vezes in range(1,25,1):
      GPIO.output(SCK,0)
      time.sleep(0.001)
      if GPIO.input(SO)
        conv = conv | aux
      aux=aux>>1
      GPIO.output(SCK,1)
      time.sleep(0.001)

   GPIO.output(CS,0)
   return(conv)

#*  *  *  *  *  *  *  *  *  *  *  *  *
#*        Função de LCD 4 vias       *
#*  *  *  *  *  *  *  *  *  *  *  *  *
def lcd_inst4(dado):
    GPIO.output(RS,0)
    if dado & 0x08:
        GPIO.output(D7,1)
    else:
        GPIO.output(D7,0)
    if dado & 0x04:
        GPIO.output(D6,1)
    else:
        GPIO.output(D6,0)
    if dado & 0x02:
        GPIO.output(D5,1)
```

```
    else:
        GPIO.output(D5,0)
    if dado & 0x01:
        GPIO.output(D4,1)
    else:
        GPIO.output(D4,0)
    GPIO.output(EN,1)
    GPIO.output(EN,0)
    time.sleep(0.01)
    return

#*  *  *  *  *  *  *  *  *  *  *  *  *
#*        Função de LCD 8 vias        *
#*  *  *  *  *  *  *  *  *  *  *  *  *
def lcd_inst8(dado):
    GPIO.output(RS,0)
    time.sleep(0.02)
    if dado & 0x80:
        GPIO.output(D7,1)
    else:
        GPIO.output(D7,0)
    if dado & 0x40:
        GPIO.output(D6,1)
    else:
        GPIO.output(D6,0)
    if dado & 0x20:
        GPIO.output(D5,1)
    else:
        GPIO.output(D5,0)
    if dado & 0x10:
        GPIO.output(D4,1)
    else:
        GPIO.output(D4,0)
    GPIO.output(EN,1)
    GPIO.output(EN,0)

    lcd_inst4(dado)
    return
```

```
#*  *  *  *  *  *  *  *  *  *  *  *  *
#*        Função de LCD DADOS         *
#*  *  *  *  *  *  *  *  *  *  *  *  *
def lcd_dado(dadostr):
    GPIO.output(RS,1)
    dado=ord(dadostr)
    if dado & 0x80:
        GPIO.output(D7,1)
    else:
        GPIO.output(D7,0)
    if dado & 0x40:
        GPIO.output(D6,1)
    else:
        GPIO.output(D6,0)
    if dado & 0x20:
        GPIO.output(D5,1)
    else:
        GPIO.output(D5,0)
    if dado & 0x10:
        GPIO.output(D4,1)
    else:
        GPIO.output(D4,0)

    GPIO.output(EN,1)
    GPIO.output(EN,0)
    if dado & 0x08:
        GPIO.output(D7,1)
    else:
        GPIO.output(D7,0)
    if dado & 0x04a:
        GPIO.output(D6,1)
    else:
        GPIO.output(D6,0)
    if dado & 0x02:
        GPIO.output(D5,1)
    else:
        GPIO.output(D5,0)
    if dado & 0x01:
        GPIO.output(D4,1)
```

```
    else:
      GPIO.output(D4,0)
    GPIO.output(EN,1)
    GPIO.output(EN,0)
    time.sleep(0.01)
    return

#*  *  *  *  *  *  *  *  *  *  *  *
#*        Função de LCD INIT        *
#*  *  *  *  *  *  *  *  *  *  *  *
def lcd_inic():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(D7,GPIO.OUT)
    GPIO.setup(D6,GPIO.OUT)
    GPIO.setup(D5,GPIO.OUT)
    GPIO.setup(D4,GPIO.OUT)
    GPIO.setup(EN,GPIO.OUT)
    GPIO.setup(RS,GPIO.OUT)
    time.sleep(1);
    lcd_inst4(0x03);
    lcd_inst4(0x03);
    lcd_inst4(0x03);
    lcd_inst4(0x02);
    lcd_inst8(0x28);
    lcd_inst8(0x0C);
    lcd_inst8(0x06);
    lcd_inst8(0x01);
    return

#*  *  *  *  *  *  *  *  *  *  *  *
#*        Função de LCD Texto       *
#*  *  *  *  *  *  *  *  *  *  *  *
def lcd_texto(linha,coluna,texto):
    coluna=coluna-1
    if linha==1:
        lcd_inst8(0x80+coluna)
    else:
        lcd_inst8(0xC0+coluna)
    for contador in range(0,len(texto)):
```

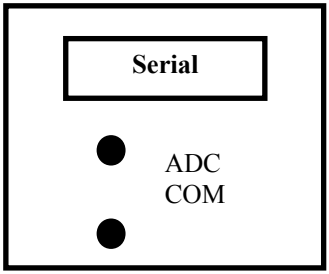
```
    lcd_dado(texto[contador:contador+1])
return
```

Capítulo IV

Amperímetro

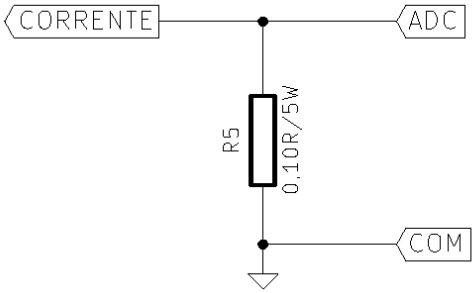
1. Introdução

Um amperímetro é um instrumento capaz de medir a corrente de um circuito e apresentá-la em uma interface homem-máquina, que pode ser do tipo display de 7 segmentos, LCD, display gráfico, etc. Estes instrumentos possuem uma baixa-impedância de entrada, para que assim não interfiram no funcionamento do circuito mensurado e podem medir ampla faixa de corrente a nível de corrente contínua (DC) ou alternada (AC). Neste capítulo, será desenvolvido um amperímetro DC utilizando uma entrada analógica, onde o resultado mensurado é enviado via porta serial, conforme sugere o modelo apresentado a seguir.



Como o amperímetro fica em série com o circuito a ser mensurado, a entrada COM representa a entrada enquanto que ADC representará a saída do medidor.

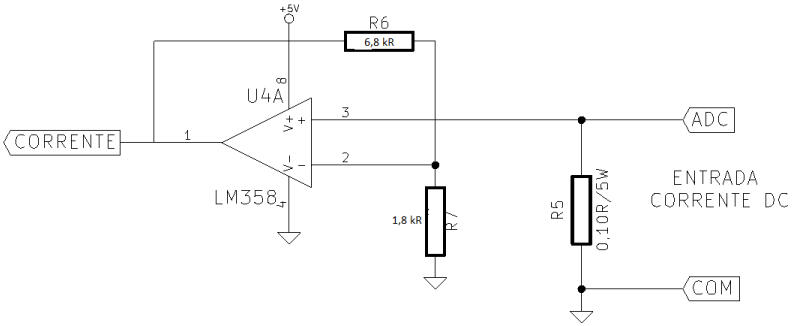
A faixa de corrente contínua projetada para medição será de 0,0 A a 7,0 A. Como a medição está na faixa de 0 V a 3,3V, é necessário um circuito de conversão corrente para tensão utilizando um resistor shunt de 0,10Ω/5W, que fornece uma tensão proporcional a corrente que passa pelo mesmo. Observe o circuito de conversão corrente-tensão a seguir que oferece o funcionamento comentado.



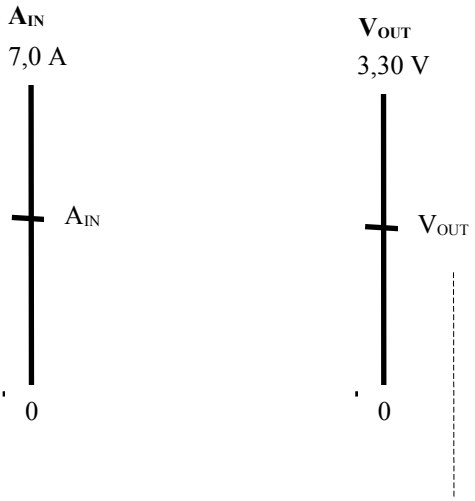
As entradas ADC e COM servem para realizar a medição no circuito enquanto que a saída CORRENTE fornece a tensão proporcional a corrente que passa pelo resistor shunt. Quando a corrente de 7 A passar pelo circuito, é observado a seguinte tensão no terminal CORRENTE, segundo a 1° Lei de Ohm.

$$V = R.I$$
$$V = 0{,}10.7$$
$$V = 0{,}70V$$

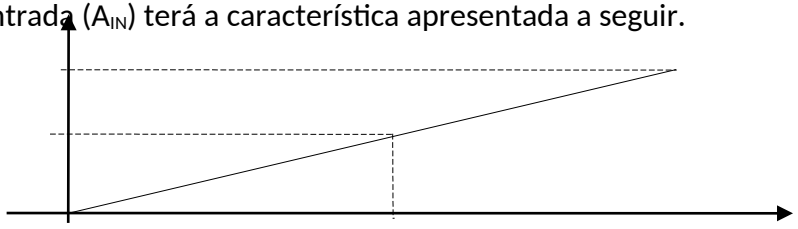
Esta tensão é muito pequena para ser tratada pelo conversor ADC que possui faixa de 0 a 3,3 V. Sendo assim é necessário amplificá-la através de um amplificador operacional (AOP) não inversor com ganho de 4,7. Isso será obtido através do circuito abaixo.

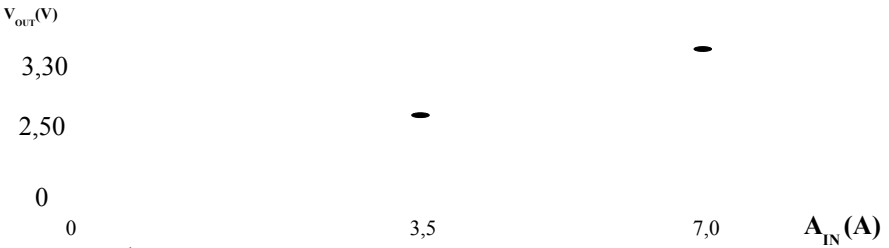


A tensão de 3,3 V será a máxima na entrada analógica. Desta forma, a figura abaixo mostra a relação da corrente medida e da entrada analógica.



A curva que relaciona a tensão de saída (V_{OUT}) com a corrente de entrada (A_{IN}) terá a característica apresentada a seguir.





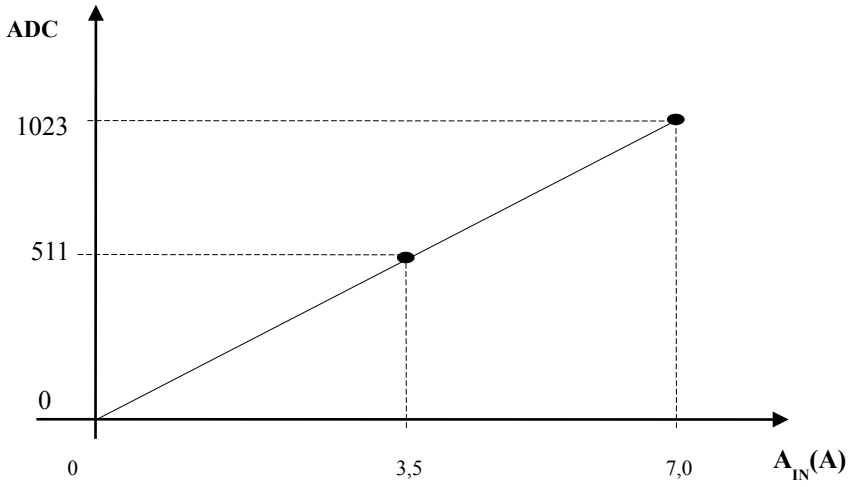
7trod
uçma

DC

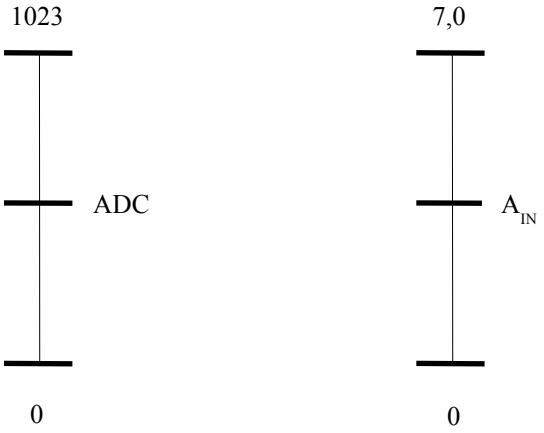
ment
oona
ment

ito o, no momento que houver 102,3 V na

O conversor ADC é de 10 bits, o que indica que o mesmo é capaz de ter uma resolução de 1024 pontos, já que 2^{10} é igual a este valor. A tensão a ser medida está ligada a uma das entradas analógicas, no qual fornece como resultado um valor binário na faixa de 0 a 1023 de acordo com a tensão de 0 a 3,3 V. Sendo assim, pode-se modelar o seguinte gráfico, no qual converte a escala de entrada para os bits referentes a conversão ADC.



Desta forma, há a seguinte relação matemática entre o resultado da conversão ADC e a medição da entrada de corrente.



$$\frac{ADC - 0}{1023 - 0} = \frac{A_{IN} - 0}{7,0 - 0}$$

$$\frac{ADC}{1023} = \frac{A_{IN}}{7,0}$$

$$A_{IN} = 7,0 \frac{ADC}{1023}$$

Ou seja, após a leitura da entrada analógica, deve-se executar o cálculo matemático acima de forma a obter o valor da corrente de entrada medida pelo circuito.