Universidade Federal de Campina Grande

Centro de Engenharia Elétrica e Informática

Curso de Graduação em Engenharia Elétrica

MARYLIA GABRIELA BARRETO BAIÉ

ESTUDO DE MÓDULOS DE COMUNICAÇÃO SEM FIO

Campina Grande, Paraíba
Março de 2017

MARYLIA GABRIELA BARRETO BAIÉ

ESTUDO DE MÓDULOS DE COMUNICAÇÃO SEM FIO

Trabalho de Conclusão de Curso submetido à Unidade Acadêmica de Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciências no Domínio da Engenharia Elétrica.

Área de Concentração: Eletrônica, controle e automação.

Orientador:
Professor José Gutembergue de Assis Lira.

Campina Grande, Paraíba
Março de 2017

MARYLIA GABRIELA BARRETO BAIÉ

ESTUDO DE MÓDULOS DE COMUNICAÇÃO SEM FIO

Trabalho de Conclusão de Curso submetido à Unidade Acadêmica de Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciências no Domínio da Engenharia Elétrica.

Área de Concentração: Eletrônica, controle e automação.

Aprovado em / /

Professor Avaliador
Universidade Federal de Campina Grande
Avaliador

Professor José Gutembergue de Assis Lira
Universidade Federal de Campina Grande
Orientador, UFCG

Dedico este trabalho aos meus pais Geovani e Marli e à minha irmã Miriam, que são o meu suporte diário e a base da minha felicidade.

AGRADECIMENTOS

Agradeço, antes de tudo e todos, a Deus, por, em sua infinita graça, ter me permitido acordar a cada novo amanhecer com a força necessária para vencer os obstáculos e as dificuldades diárias.

Aos meus pais Marli e Geovani, que estão comigo desde os primeiros passos, a quem tudo devo, que são minha fortaleza e meus exemplos maiores de perseverança; pelo apoio incondicional e pelo incentivo de sempre, me motivando continuamente a ser o meu melhor. A minha irmã Miriam, eterna amiga, que sempre está ao meu lado, me ajudando nos momentos que mais preciso. E também a meus avós, tios e primos que enchem minha vida de alegria.

À UFCG, por me permitir crescer intelectualmente, oferecendo um curso de excelência em minha área, construído por um corpo docente extremamente qualificado, essencial para minha formação profissional, e por uma equipe de funcionários administrativos que facilitaram o percurso que me permitiu chegar até aqui.

Ao professor Dr. Gutembergue de Assis Lira, por ter sido um grande mestre durante o curso, por ter aceitado ser meu orientador, por acreditar sempre no meu potencial e por continuamente me entusiasmar com a profissão.

Aos meus amigos que me acompanham desde a época de escola, Kleanny, Priscila, Karol, Rodolpho e Mauro. A amizade de vocês sempre foi um suporte essencial em toda minha trajetória acadêmica e pessoal.

A todos que já foram meus professores, cada um foi de fundamental importância para minha trajetória profissional.

Por fim, agradeço àqueles colegas de curso, que me acompanharam e me ajudaram durante toda minha trajetória de graduação, sem eles o percurso teria sido muito mais árduo.

"Não vos esforceis pelas honras
do mundo, mas honrai o Senhor."

São Francisco de Assis.

RESUMO

A palavra *wireless* é um termo em inglês cujo significado é “sem fio” (*wire* - fio, *less* - sem ou menos). Nos últimos anos, o uso de redes sem fio tem se popularizado, a ideia desse tipo de comunicação, no entanto, não é tão recente. A popularização da comunicação sem fio abrange todos os meios, sejam eles industriais, comerciais ou residenciais. Neste último, em especial, as aplicações de automação tem crescido fortemente. Atualmente, existe no mercado uma grande variedade de módulos que cumprem essa função de “transportar” informações à distância. Este trabalho trará o estudo de alguns desses módulos, suas características e como funcionam, além de alguns testes e aplicações que sirvam de base àqueles que desejam utilizá-los. Também trará o estudo sobre os componentes eletromecânicos utilizados nestes testes. Estes módulos necessitam, normalmente, de um microcontrolador para interpretar as informações recebidas ou para, em muitos casos, organizar as informações que serão enviadas. Para tal, foi utilizada a placa de desenvolvimento Arduíno. A escolha para a utilização de cada módulo deve ser feita de acordo com as necessidades da aplicação desejada.

Palavras-chave: *wireless*, automação, comunicação, distância, módulos, Arduíno.

ABSTRACT

The word wireless is an English term whose meaning is "without wire". In recent years, the use of wireless networks has become popular, the idea of this type of communication, however, is not so recent. The popularization of wireless communication covers all fields, such as industrial, commercial or residential. The latter, in particular, automation applications have grown strongly. Currently, there is a wide variety of modules on the market that fulfill this function of "transporting" information from the distance. This work approached some of these modules, their characteristics and how they work, as well as some tests and applications to serve as a basis for those who wish to use them. It will also bring the study of electromechanical components used for these tests. These modules usually require a microcontroller to interpret the received information or, in many cases, to organize the information that will be sent. For this, the Arduino development board was used. The choice in each module should be made according to the needs of the desired application.

Keywords: Wireless, automation, communication, distance, modules, Arduino.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AC	*Alternative current* (corrente alternada)
ADC	*Analog to digital converter*
AM	*Amplitude modulation*
AT	*Attention*
BLE	*Bluetooth Low Energy*
Bps	*Byte per second*
bps	*Bits per second*
BR	*Basic Rate*
BT	*Bluetooth*
CI	Circuito Integrado
CPU	*Central Process Unit*
CRC	*Cyclic redundancy check*
DC	*Direct current* (corrente contínua)
DQPSK	*Differential quadrature phase shift keying*
DTR	*Data terminal ready*
E/S	Entrada/ Saída
EDR	*Enhanced Data Rate*
FM	*Frequency Modulation*
FTDI	*Future Technology Devices International*
GFSK	*Gaussian frequency shift keying*
GND	*Ground*
GPIO	*General Purpose Input/output*
I/O	*Input/ Output*
ICSP	*In Circuit Serial Programming*
IDE	*Integrated Development Environment*
IEM	Interferência eletromagnética
ISM	*Industrial*, *Scientific*, *Medical*
MCU	Microcontrolador
NA	Normalmente aberto
NF	Normalmente fechado

PROM	*Programmable Read Only Memory*
PSK	*Phase-shift keying*
PWM	*Pulse Width Modulation*
RAM	*Random Access Memory*
RF	Rádio Frequência
RFI	Interferência de rádio frequência
RX	Receptor
SPI	*Serial Peripherical Interface*
SSR	*Solid State Relay*
TTL	*Trasnsistor – transistor logic*
TX	Transmissor
UART	*Universal Asynchronous Receiver/Transmitter*

LISTA DE SÍMBOLOS

μ	Micro (10^{-6})
A	Ampére
B	Byte
dBm	Decibel miliwatt
G	Giga (10^{9})
Hz	Hertz
K	Kilo (10^{3})
m	Mili (10^{-3})
M	Mega (10^{6})
n	Nano (10^{-9})
V	Volt
W	Watts
Ω	Ohm

SUMÁRIO

1 INTRODUÇÃO

Com a crescente evolução da tecnologia, de uma forma geral, o mundo está se tornando cada vez mais móvel. A necessidade humana de eliminar fios e ter mobilidade – quer nas mais variadas formas de comunicação, quer nas formas de acesso à informação ou automação de processos – é hoje uma realidade que norteia o rumo da tecnologia (Sampaio, 2008).

A atividade do homem atual vem sendo facilitada por processos cada vez mais automatizados, e isto é possível de acontecer devido aos microcontroladores que agem em conjunto com equipamentos que permitem a comunicação sem fio.

Wireless origina-se do inglês e significa: sem fio (*wire* = fio, cabo; *less* = sem). Sendo assim, especifica qualquer tipo de conexão para transmissão e recepção de informação sem a utilização de fios ou cabos. São inúmeros os módulos associados aos microcontroladores que suprem essa função, tão presente não só mais na indústria, mas também no dia-a-dia do homem atual.

Os processos automatizados oferecem soluções cada vez mais abrangentes até mesmo quando em aplicações *wireless*, onde as distâncias físicas entre o transmissor e o receptor são bastante notáveis. Para tal, cada sistema desses processos, possui exigências específicas da aplicação para a qual foi idealizada.

Para que um projeto de automação, ou qualquer outro que necessite de uma comunicação sem fio, funcione adequadamente, é necessário escolher todos os componentes que o compreendem de forma sensata à aplicação que se deseja utilizar. Sendo assim, este trabalho teve o intuito de apresentar algumas opções de circuitos que proveem a função de transmitir informações sem fio, visto que no mercado são ofertadas inúmeras opções para tal função. Cada módulo apresentado terá definido suas características distintas, de operação e funcionamento, testes, com estes associados a microcontroladores, para acionamento de cargas e monitoramento de sistemas. Assim, de acordo com as aplicações que os demandem, a escolha entre eles possa ser realizada de forma eficiente.

2 MICROCONTROLADORES

Microcontroladores são geralmente utilizados em automação e controle de produtos ou processos completos, como por exemplo, processos de pintura de produtos em uma indústria, sistemas de controle automotivos, sistemas de segurança, máquinas residenciais, brinquedos, sistemas de supervisão, entre outros. Por reduzir o tamanho, ter seu custo e consumo de energia cada vez menores, aliados as facilidades de implementação, os microcontroladores são uma alternativa eficiente para controlar muitos processos e aplicações.

Um microcontrolador é um Circuito Integrado (CI) capaz de efetuar processos lógicos. A grande vantagem deste CI é a sua possibilidade de programação, o que o torna adaptável à finalidade desejada, e que possibilita seu ajuste de acordo com a tarefa que deverá executar (Sousa D. J., 2000).

O desenvolvimento da tecnologia de circuitos integrados, que tornou possível o armazenamento de milhares de transistores em um único chip, suscitou a produção de microprocessadores. Estes, aliados a periféricos externos, como memórias, conjuntos de I/O, temporizadores, entre outros, fizeram emergir os computadores. A crescente necessidade de busca pela redução dos tamanhos físicos de componentes e equipamentos com o passar do tempo, desencadeou a integração entre periféricos e processador em um único circuito integrado, que passou a ser designado de microcontrolador (MCU) (Assis, 2004).

O microcontrolador possui, basicamente, os seguintes dispositivos:

- CPU (*Central Processing Unit*), cuja finalidade é interpretar as instruções de programa;
- Memória PROM (*Programmable Read Only Memory*), na qual são gravadas as instruções do programa;
- Memória RAM (*Random Access Memory*), utilizada para memorizar as variáveis utilizadas pelo programa;
- Conjunto de entradas e saídas para controlar dispositivos externos ou receber sinais de sensores, interruptores, entre outros;
- Conjuntos de dispositivos auxiliares ao funcionamento, ou seja, gerador de *clock*, contadores, unidades para comunicação, conversores entre outros.

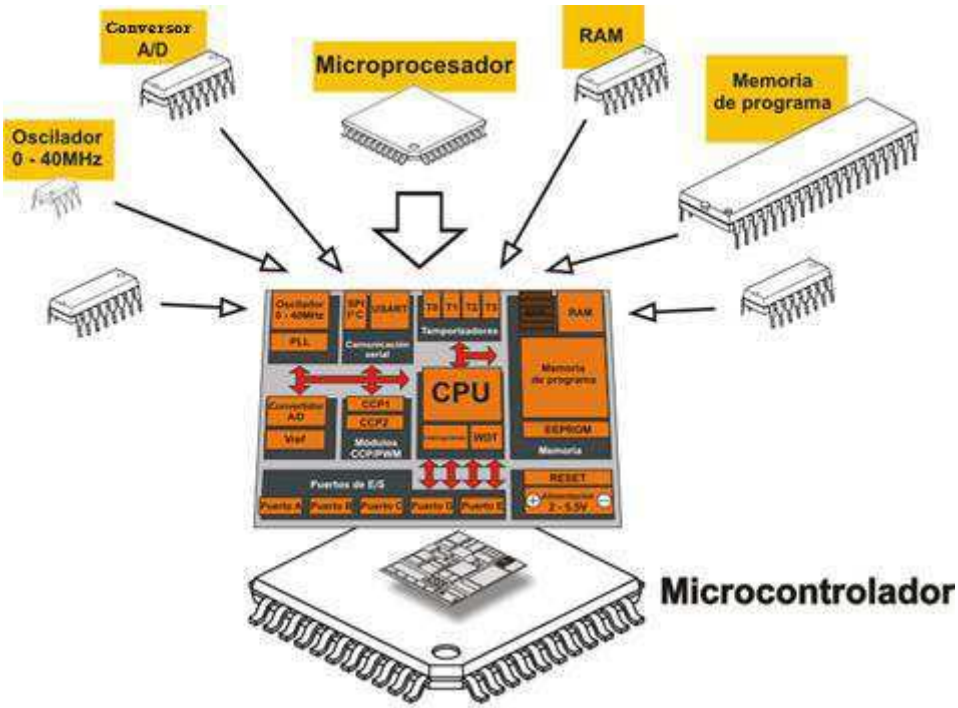


Figura 1 - Composição de um microcontrolador[1].

Os microcontroladores não possuem, geralmente, um sistema operacional. Os programas rodam diretamente no chip. O *software* que roda no microcontrolador é denominado *Firmware*. Esse *software* é programado em linguagens C ou Assembly na maior parte dos casos, embora seja possível usar outras linguagens em alguns MCUs. A programação é feita com o uso de uma ferramentas instalada em um computador, chamada de IDE (do inglês *Integrated Development Environment* ou Ambiente de Desenvolvimento Integrado).

No mercado, são diversas as empresas que fabricam microcontroladores, quais sejam: Mitsubishi, Atmel, Freescale, Texas Instruments, Intel, Microchip Technology, entre outras. Os tipos destes chips variam de acordo com a aplicação, a velocidade de operação, a capacidade de memória, o tipo de memória, tamanho da palavra, entre outros. Alguns exemplos típicos de microcontroladores comuns encontrados no mercado e usados comumente em projetos eletrônicos estão listados a seguir:

- ARM Cortex-M;
- Atmel AVR / AVR 32;

[1] Mikroe. (s.d.). *MikroElektronika - Development tools*. Acesso em 10 de Março de 2017, disponível em www.learn.mikroe.com: https://learn.mikroe.com/ebooks/microcontroladorespicbasic/chapter/introduccion-al-mundo-de-los-microcontroladores/

- Intel 8051;
- Microchip PIC;
- NXP LCP 2000 / 3000;
- Paralax Propeller;
- Texas Instruments MSP430.

É também muito comum no mercado kits e placas para estudo e prototipagem de sistemas embarcados e programação de microcontroladores. Como exemplo, cita-se a Plataforma Arduíno, o Microchip PIC Starter Kit, o Texas Instruments MSP430 Launchpad e o Keil MCB900 (NXP). Estes facilitam o desenvolvimento de um equipamento, como por exemplo, em testes dos circuitos associados aos microcontroladores.

2.1 PLACA DE DESENVOLVIMENTO ARDUÍNO

Arduíno é uma plataforma para prototipagem eletrônica que utiliza o conceito de *hardware* livre. Neste conceito, deve-se prover a disponibilização irrestrita de informações sobre o projeto de *hardware*, tais como diagramas eletrônicos, estrutura de produtos, *layout* de placas de circuito impresso, rotinas de baixo nível e qualquer outra informação necessária para uma construção de um projeto (Silva & Sales, 2016).

A ideia inicial de criação da placa (que foi desenvolvida no *Ivrea Interaction Design Institute*) foi de um instrumento para prototipagem simples de sistemas para aqueles que não possuíam conhecimentos aprofundados nas áreas de eletrônica e programação. À medida que foi se popularizando para uma comunidade mais especializada, a placa foi adquirindo adaptações que a tornou favorável a aplicações mais sofisticadas, como para IoT, impressão 3D, sistemas embarcados, automações residenciais e *wearables*.

Ao longo dos anos, o Arduíno tem sido o núcleo de milhares de projetos, desde objetos cotidianos até complexos instrumentos científicos. Uma comunidade mundial de criadores - estudantes, amadores, artistas, programadores e profissionais - se reuniram em torno desta plataforma de *hardware* e *software* aberto e suas contribuições somaram uma incrível quantidade de conhecimento acessível[2].

[2] Arduíno. (2017). *Arduíno - What is Arduino?* Acesso em 15 de Janeiro de 2017, disponível em www.arduino.cc: https://www.arduino.cc/en/Guide/Introduction

O Arduíno também simplifica o processo de trabalhar com microcontroladores como em muitas outras plataformas, além de oferecer algumas vantagens em comparação a outros:

- Baixo custo - as placas são relativamente baratas em comparação com outras plataformas de microcontrolador;
- *Cross-platform* (Multiplataforma) - O *software* Arduíno (IDE) é executado em sistemas operacionais Windows, Macintosh OSX e Linux. A maioria dos sistemas de microcontroladores é limitada ao Windows;
- Ambiente de programação simples e claro - O *software* Arduíno (IDE) é fácil de usar para iniciantes, mas flexível o suficiente para que os usuários avançados também possam aproveitar;
- *Software open source* e extensível - O *software* Arduíno é publicado como ferramentas de código aberto, disponíveis para a extensão por programadores experientes. O idioma pode ser expandido por meio de bibliotecas C ++, e as pessoas que querem entender os detalhes técnicos podem fazer o salto do Arduíno para a linguagem de programação AVR C em que se baseia. Da mesma forma, se desejar, pode adicionar códigos AVR-C diretamente em seus programas;
- *Hardware open source* e extensível - Os planos das placas Arduíno são publicados sob uma licença *Creative Commons*, para que designers de circuitos experientes possam fazer sua própria versão do módulo, estendê-la e melhorá-la. Mesmo os usuários relativamente inexperientes podem construir a versão de painéis do módulo, a fim de entender como ele funciona e economizar dinheiro.

Existe uma gama de placas Arduíno no mercado, cada uma com diferentes especificações (o microcontrolador, quantidade de portas e conexões, memória, entre outros). A escolha dependerá sempre da aplicação a qual a placa será destinada. A Figura 2 apresenta um resumo dos existentes atualmente.

Nos tópicos subsequentes são apresentadas as características do Arduíno UNO e do Arduíno mini, que foram os utilizados neste projeto.

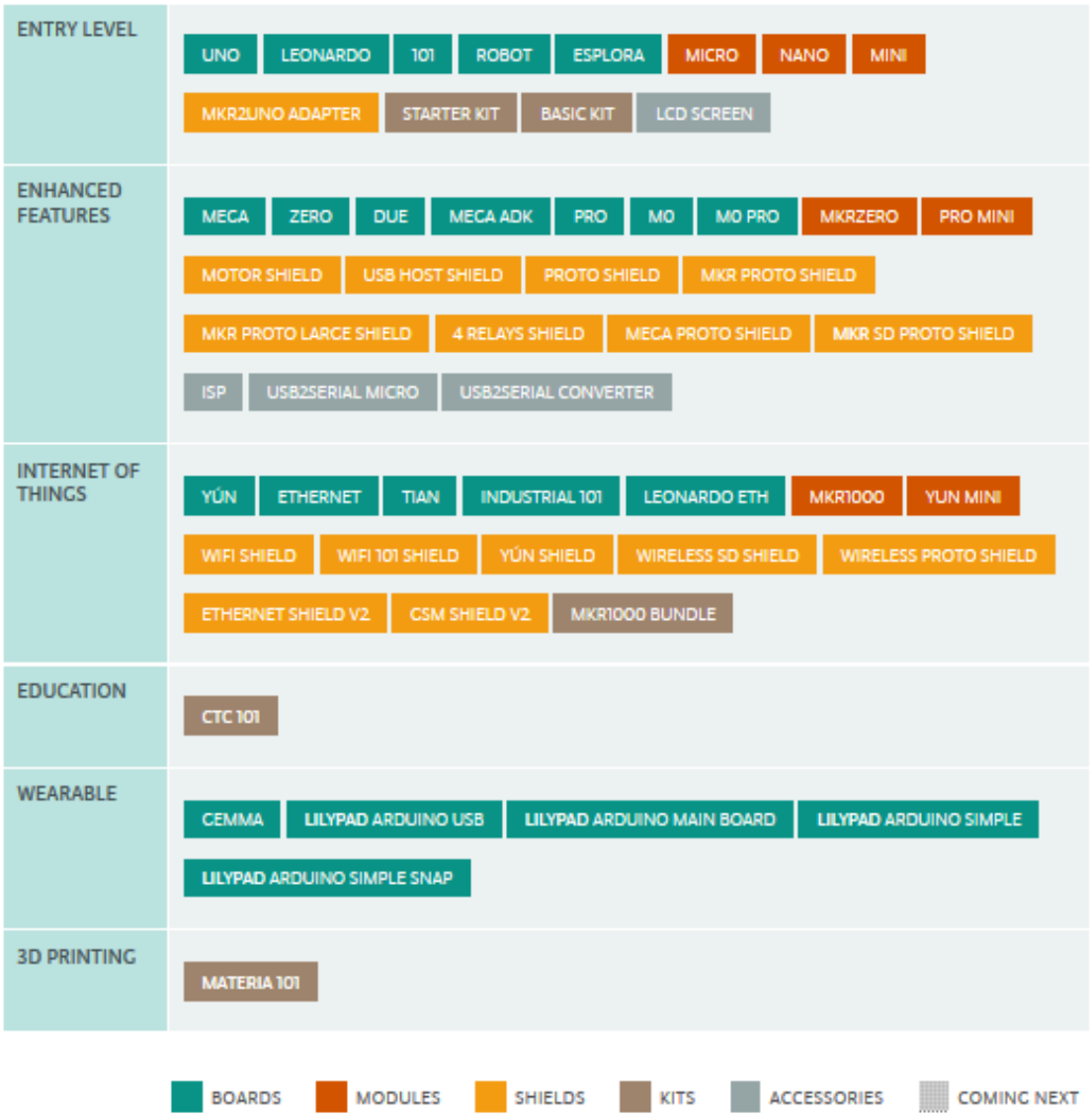


Figura 2 - Tipos de placa Arduíno[2].

2.1.1 ARDUÍNO UNO

O UNO possui o processador ATMEGA 328, 14 portas digitais, sendo que seis delas podem ser usadas como saídas PWM, e 6 portas analógicas. A alimentação (selecionada automaticamente) pode vir da conexão USB ou do conector para alimentação externa (recomendável 7 a 12 Vdc). Possui uma tensão de operação de 5 V, cristal oscilador de 16 MHz, corrente máxima nas portas de E/S de 40 mA, uma conexão ICSP (*In Circuit Serial Programming*, um protocolo de comunicação utilizado para gravação de dispositivos programáveis) e um botão de reset. Em relação à memória, o mesmo possui as seguintes características (todos do Atmega 328):

- Memória flash de 32 KB dos quais 0,5 KB é utilizada pelo bootloader;

- SRAM de 2 KB;
- EEPROM de 1 KB.

O Arduíno possui um *firmware* chamado *bootloader* que é executado ao alimentar a placa ou quando pressionado o botão de *reset*. Este tem a função de carregar um programa do computador escrito na IDE, e escrever na memória do microcontrolador. Este *firmware* é responsável pela facilidade que o Arduíno oferece de ser programado diretamente pela porta serial (que na realidade ocorre pela USB, o que acontece é que existe outro microcontrolador nas placas do Arduíno UNO, conversor serial-USB, que cria uma porta serial virtual no computador quando o Arduíno é conectado).

Cada um dos 14 pinos digitais do UNO pode ser utilizado como uma entrada ou uma saída utilizando-se as funções pinMode(), digitalWrite(), e digitalRead(). Eles possuem um resistor *pull-up* interno (desconectado por padrão) de 20-50 kΩ. Além disso, alguns pinos têm funções especializadas:

- Serial: 0 (RX) e 1 (TX). Usados para receber (RX) e transmitir (TX) dados seriais TTL. Estes pinos são conectados aos pinos correspondentes do chip, conversor, serial USB-para-TTL ATmega8U2 ;
- Interruptores Externos: 2 e 3. Estes pinos podem ser configurados para disparar uma interrupção de acordo com alguma variação sensível pelo circuito (função attachInterrupt());
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Estes pinos dão suporte à comunicação SPI utilizando a biblioteca SPI;
- LED: 13. Há um LED integrado ao pino digital 13;
- I2C: 4 (SDA) e 5 (SCL). Fornecem suporte à comunicação I2C (TWI) utilizando a biblioteca Wire;
- AREF: Tensão de referência para as entradas analógicas. Utilizado com a função analogReference ();
- Reset: Envio o valor LOW para esta linha para resetar o microcontrolador. Tipicamente usado para adicionar um botão de reset para *shileds* montados sobre a placa original.

As entradas analógicas, A0 a A5, tem 10 bits de resolução cada (variam então com valores de 0 a 1023). Por padrão elas medem de 0 a 5 V, embora seja possível

alterar o limite superior utilizando o pino AREF e a função analogReference (). O anexo A apresenta o mapa de pinos de toda a placa.

2.1.2 ARDUÍNO PRO-MINI

Existem duas versões do Arduíno Pro Mini, uma baseada no microcontrolador Atmega 328, executado em 5 V e 16 MHz e outra no Atmega 168, executado em 3.3 V e 8 MHz. Contudo, neste trabalho será caracterizada e utilizada a versão com o Atmega 328

Esta placa também possui 14 pinos de E/S digitais, dos quais seis podem ser usados como saídas PWM, oito entradas analógicas e um botão de *reset*, como no UNO. No entanto, não possui conexão USB ou um conector direto para alimentação externa. Para tal, pode ser utilizado um cabo FTDI (conversor USB para Serial (nível TTL)) ou uma placa externa.

Se a necessidade for efetivar apenas a alimentação da placa, por uma placa externa deve-se conectar uma alimentação regulada de 3.3V ou 5 V (dependendo do modelo) ao pino Vcc. Há também um regulador de tensão na placa que suporta uma tensão de até 12 VDC, neste caso deve-se conectar ao pino "RAW" e não ao “Vcc”.

O Arduíno Pro Mini destina-se à instalação semipermanente ou permanente em objetos ou qualquer que seja a aplicação. A placa vem sem os pinos pré-montados, permitindo o uso de vários tipos de conectores ou soldagem direta de fios.

Suas dimensões são 33 x 18 x 6 mm e as especificações de memória e dos pinos de E/S são as mesmas do UNO. O mapa de pinos do mesmo pode ser visualizado no Anexo B.

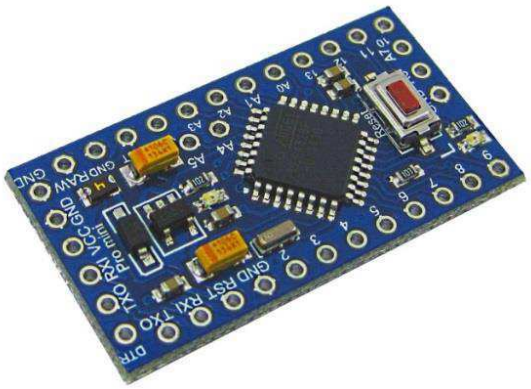


Figura 3 - Arduíno Pro-Mini.

3 COMUNICAÇÃO POR RÁDIO FREQUÊNCIA

Uma conexão *wireless* é qualquer forma de conexão entre dois sistemas-transmissor e receptor de dados -, que não requeira o uso de fios, como já apresentado.

Em um sistema de comunicação sem fio que utilize ondas de rádio, a informação a ser transmitida é modulada em uma portadora, ou seja, ela é posicionada no espectro de frequências de modo que no mesmo meio físico possam trafegar informações de vários transmissores, desde que estejam utilizando uma faixa não ocupada. Por meio da modulação é possível fazer o deslocamento do espectro da informação para outra região não ocupada. O fato de não existirem fios ligando os dispositivos de comunicação permite que estes dispositivos ofereçam mobilidade. Para explorar esta vantagem, torna-se vital o baixo consumo nos transmissores/receptores para que estes possam ser alimentados com baterias pequenas (Sousa M. B., 2002).

Os sinais de radiofrequência são sinais que se propagam por um condutor e são irradiados no ar por meio de uma antena. Na prática, uma antena converte um sinal cabeado em um sinal *wireless* e vice-versa. Esses sinais são então irradiados no ar livre na forma de ondas de rádio e se propagam em linha reta e em todas as direções. Podem-se imaginar essas ondas como círculos concêntricos que aumentam o seu raio na medida em que se afastam da antena (Sampaio, 2008).

Para implementar um sistema de transmissão de dados por ondas de radiofrequência, necessita-se principalmente de módulos de radiofrequência, responsáveis pela transmissão e recepção dos dados via ondas de rádio. Este é o tema central deste trabalho, e a partir dos próximos itens são retratados os módulos que, associados a microcontroladores (no caso, o Arduíno), cumprem essa aplicação de enviar dados sem fio para o acionamento de cargas ou monitoramento de sistemas e os tipos de comunicação serial que os mesmos utilizam.

Estes módulos que serão apresentados atuam como meios de comunicação capazes de pôr em prática diversas ações, como funções de telemetria (efetivar medidas de sensores à distância), acionamento de cargas por meio do acionamento direto de relés ou por acoplamento óptico, envio e recepção de dados analógicos e/ou digitais para as mais variadas aplicações, entre outras.

3.1 COMUNICAÇÃO SERIAL

A comunicação do tipo serial é o processo de enviar dados, um bit de cada vez, sequencialmente, num canal de comunicação ou barramento. Já a comunicação paralela é o processo de enviar dados em que todos os bits de uma palavra são enviados juntos, ao mesmo tempo. A comunicação paralela implica mais de um fio, além da conexão de alimentação.

Diversas tecnologias de interligação serial entre dispositivos foram desenvolvidas, podendo ser separadas em duas grandes categorias, a comunicação síncrona e a comunicação assíncrona. A Figura 4, a seguir, apresenta essas categorias.

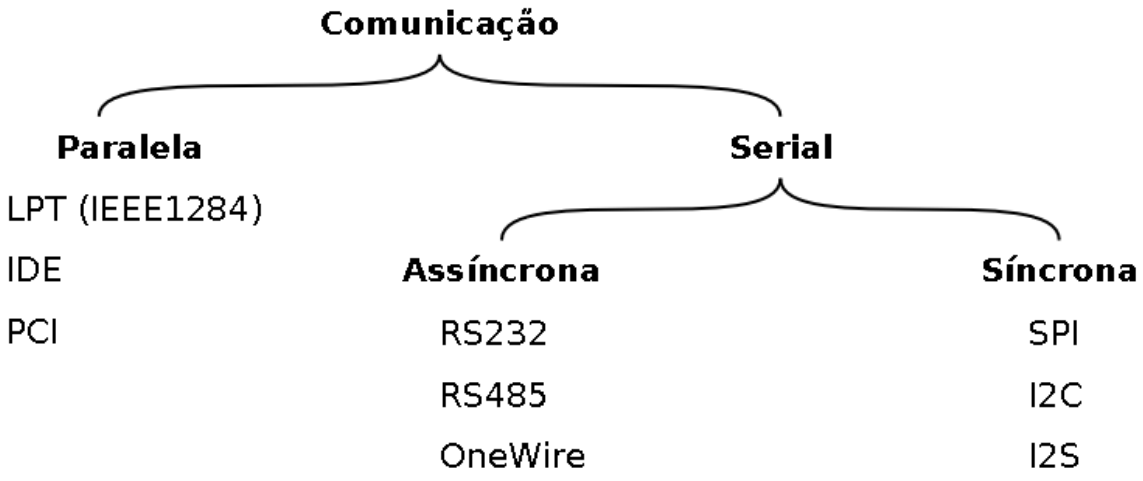


Figura 4 - Tipos de comunicação[3].

Na transmissão assíncrona, um elemento de sinalização é inserido no início e no fim da transmissão de um caractere e, assim, permite que o receptor entenda o que foi realmente transmitido. Portanto, não há coordenação antes da transmissão de um caractere. O sincronismo é mantido apenas dentro de cada caractere (Azevedo, 2010). Cada bloco de dados possui uma *flag* (espécie de controle) que informa onde começa e onde acaba esse bloco, além da posição na sequência de dados transmitida.

A transmissão síncrona tem a característica de transmitir um bloco inteiro, sem intervalos de tempo entre outros blocos. O dispositivo emissor e o dispositivo receptor devem estar em sincronia antes da comunicação iniciar e manter esta sincronia durante

[3] Sacco, F. (05 de maio de 2014). *Embarcados.* Acesso em 10 de março de 2017, disponível em www.embarcados.com.br: https://www.embarcados.com.br/spi-parte-1/

toda a transmissão. Cada bloco de informação é transmitido e recebido num instante de tempo bem definido e conhecido pelo transmissor e receptor. Quando um bloco é enviado, o receptor é bloqueado e só pode enviar outro bloco quando o primeiro for recebido pelo receptor.

A seguir são apresentadas com mais detalhes as comunicações seriais do tipo RS232 e a SPI, que são as utilizadas nos módulos apresentados mais a adiante.

3.1.1 PROTOCOLO RS232

RS é uma abreviação de "*Recommended Standard*". Ela relata uma padronização de uma interface comum para comunicação de dados entre equipamentos, criada no início dos anos 60, por um comitê conhecido atualmente como "*Electronic Industries Association*" (EIA). Naquele tempo, a comunicação de dados compreendia a troca de dados digitais entre um computador central (*mainframe*) e terminais de computador remotos, ou entre dois terminais sem o envolvimento do computador. Estes dispositivos poderiam ser conectados através de linha telefônica e, consequentemente, necessitavam um modem em cada lado para fazer a decodificação dos sinais. Dessas ideias nasceu o padrão RS232. Ele especifica as tensões, temporizações e funções dos sinais, um protocolo para troca de informações e as conexões mecânicas (Canzian, 2009).

O equipamento que faz o processamento do sinal é chamado DTE (terminal de dados, *Data Terminal Equipament*), que normalmente é um computador, já o equipamento que faz a conexão é denominado de DCE (comunicador de dados, *Data Comunication Equipament),* normalmente um modem.

Apesar desta porta hoje perder grande parte do seu mercado para a USB (*Universal Serial Bus*), é importante entender como a mesma funciona. Ela está disponível nos computadores, ou CLPs (onde é bem mais comum encontrá-la nos dias atuais), no conector do tipo DB9 ou DB25 sendo ambos do tipo macho, sendo o tipo DB9 mais usado no dia-a-dia. A pinagem interna do DB9 pode ser visualizada na Figura 5.

Destes pinos, somente o 2, 3 e 5 são utilizados para prover comunicação entre dispositivos ficando o restante para o controle do tráfego de dados. Um dos parâmetros dessa comunicação é o *baud rate,* que informa quantos bits no período de 1 segundo serão transferidos na linha. *Baud rates* comuns são o 2400, 4800 e 9600 bps.

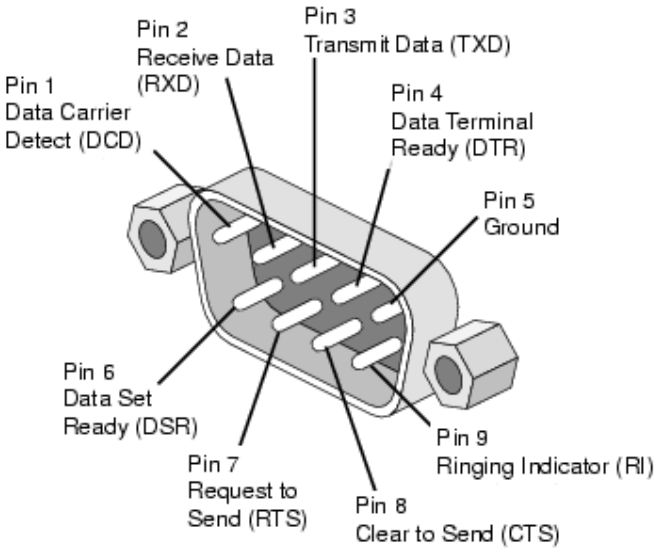


Figura 5 - Cabo DB9.

Em todos os testes deste trabalho foi utilizado a conexão USB quando necessária a comunicação serial RS232 por meio dos módulos ou da placa Arduíno. Para os módulos de comunicação serial fez-se necessário o uso de conversores USB-serial (que será apresentado na seção 4.4), e para as placas Arduíno, as mesmas já possuem este conversor integrado.

3.1.2 PROTOCOLO SPI

Na comunicação SPI (*Serial Peripherical Interface*) os sinais de comunicação possuem uma direção fixa e definida. Sempre existem dois transistores definindo o estado de um pino (*Push-Pull*). Essa característica é uma das grandes diferenças entre outras comunicações seriais como na I2C e *OneWire*, que possuem um mesmo barramento de dados para os sinais de entrada e saída através do esquema de dreno-aberto (*Pull-Up*), como apresentado na Figura 6.

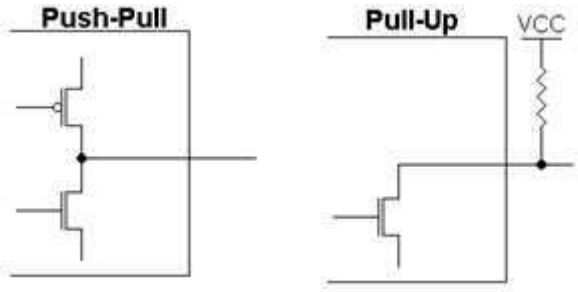


Figura 6 - Push-Pull e Pull-Up[3].

O barramento funciona em uma arquitetura mestre-escravo, em que o mestre controla todas as comunicações efetuadas, e toda troca de dados acontece sempre em ambas as direções, cada bit trocado entre o *Master* e um *Slave* traz um bit do *Slave* para o *Master*. Dessa forma, define-se que a comunicação é sempre *full-duplex*.

A comunicação é constituída por quatro conexões: SCK (*Serial Clock*), MOSI (*Master Output Slave Input*), MISO (*Master Input Slave Output*) e SS (*Slave Select*).

A conexão SCK é utilizada para transmitir o sinal de *clock* que vai controlar a velocidade com que se dará a comunicação, sendo este enviado pelo mestre. A conexão MOSI é utilizada para a transmissão de dados do mestre para o escravo e a MISO é utilizada para permitir o envio de dados no sentido inverso.

Por último, a linha de *Slave Select* é usada pelo mestre para selecionar o dispositivo com que se pretende comunicar. A seleção do respectivo escravo faz com que, a partir desse momento, apenas este leia o conteúdo da linha MOSI e se necessário envie dados através da linha MISO.

O primeiro passo para iniciar uma comunicação é configurar a frequência do sinal de *clock* para a frequência desejada. As frequências utilizadas podem atingir os 70 MHz. Depois de configurado o sinal de *clock*, o mestre deverá selecionar o dispositivo com que pretende comunicar, colocando a respectiva linha de SS a zero (nível baixo). A partir deste momento, é iniciada a transmissão dos dados, sendo possível, durante um ciclo de *clock*, a transmissão de dados tanto pela linha MOSI como também pela linha MISO, havendo assim uma comunicação em modo *full-duplex* (Ferreira, 2008). A Figura 7 apresenta esse conceito.

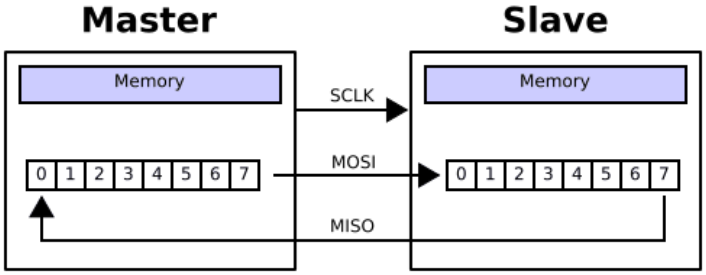


Figura 7 - Comunicação SPI.

A comunicação neste barramento pode ser descrita como uma troca de conteúdos entre dois *shift registers*, ligados em anel. Ao longo da transmissão dos dados, há uma troca progressiva entre os dois conteúdos. No final de cada transmissão, cada um dos dispositivos envolvidos lê o conteúdo do seu *buffer,* repetindo o processo

se existirem mais dados a transmitir. Se por outro lado, não houver mais dados a ser transmitido o mestre termina o envio do sinal de *clock* e "desseleciona" o escravo.

Por padrão, a comunicação SPI permite a configuração das bordas de comunicação do *clock* através de sua polaridade e fase. A configuração da polaridade se dá através de CPOL (*Clock Polarity*), que determinará a forma como será gerado o sinal de *clock* e a configuração da fase se dá através de CPHA (*Clock Phase*) que definirá o momento em que se deverão escrever e ler os dados. Seus modos possíveis estão apresentados na Tabela 1

Tabela 1- Modos de configuração do Clock.

Modo	CPOL	CPHA	Borda de troca
0	0	0	Subida
1	0	1	Descida
2	1	0	Subida
3	1	1	Descida

. Por exemplo, para CPOL = 0 e CPHA = 0, os dados serão lidos na transição de subida do sinal do *clock* e devem ser escritos na borda de descida do *clock*.

Como foi dito anteriormente, o barramento SPI permite a conexão de vários escravos ao barramento. Normalmente, a topologia usada para fazer esta conexão é utilizar sinais de SS (*Slave Select*) independentes para cada um dos escravos, controlados pelo mestre. Esta solução implica que o pino onde é ligada a linha MISO nos escravos seja de alta impedância, visto que esta linha é partilhada por todos os escravos.

A partir das seções seguintes, serão apresentados os módulos estudados, suas características e como funcionam.

3.2 MÓDULOS RF 433MHZ (MX-FS-03V E MX-05V)

Esses módulos são componentes básicos para comunicação via rádio frequência, presente em sistemas de alarmes, controle remoto, aquisição de dados e robótica em geral. A transmissão tem sentido unidirecional, não havendo retorno do receptor. Os módulos alcançam de 20 a 200 metros sem obstáculos (a adição da antena é essencial se há necessidade de se chegar ao alcance máximo), com modulação AM e frequência de trabalho de 433 MHz (no Brasil essa frequência é permitida para uso livre pela Anatel).

Outro detalhe importante é que o módulo transmissor aceita tensões na faixa de 3,5 a 12 V, o que também faz uma grande diferença no alcance da transmissão. A Figura 8 apresenta o detalhe da pinagem dos dois módulos (o transmissor e o receptor) e como devem ser realizadas suas conexões.

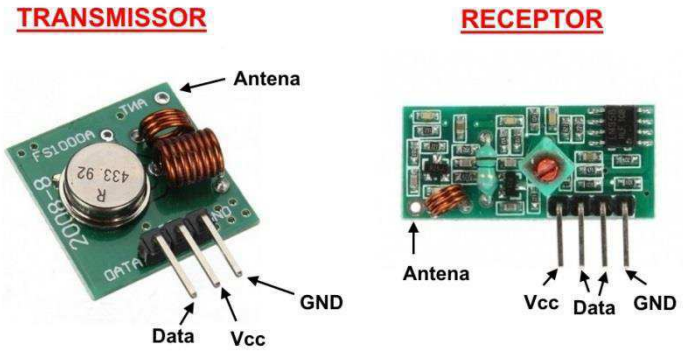


Figura 8 - Módulo RF 433MHz.

Outras especificações dos módulos são apresentadas a seguir:

Transmissor:

- Taxa de transferência: 4 KB/s;
- Potência de transmissão: 10 mW;
- Dimensões: 19 x 19 mm.

Receptor:

- Corrente de operação: 4 mA;
- Sensibilidade: -105 dBm;
- Dimensões: 30 x 14 x 7 mm.

3.3 MÓDULOS *BLUETOOTH*

3.3.1 TECNOLOGIA *BLUETOOTH*

A tecnologia Bluetooth é de curto alcance, cujo objetivo é eliminar os cabos nas conexões entre dispositivos eletrônicos, tanto portáteis como fixos. Dentre as principais características desta tecnologia, elenca-se: confiabilidade, baixo consumo e mínimo custo. Várias das funções das especificações são opcionais, o que permite a diversificação dos produtos.

Os dispositivos *Bluetooth* operam na faixa ISM (*Industrial*, *Scientific*, *Medical*) centrada em 2,45 GHz em sua camada física de Radio (RF) que era formalmente reservada para alguns grupos de usuários profissionais. Nos Estados Unidos, a faixa ISM varia de 2400 a 2483,5 MHz. Na maioria da Europa a mesma banda também está disponível. No Japão a faixa varia de 2400 a 2500 MHz.

A faixa ISM é aberta e assim, pode ser utilizada por qualquer sistema de comunicação. Portanto, é imprescindível garantir que o sinal do Bluetooth não sofra interferência e também não a gere. O esquema de comunicação FH-CDMA (*Frequency Hopping — Code-Division Multiple Access*), utilizado pelo Bluetooth, permite essa proteção, dividindo a frequência em vários canais.

O dispositivo que estabelece a conexão muda de um canal para outro de maneira bastante rápida. Esse "salto de frequência" permite que a largura de banda da frequência seja muito pequena, diminuindo sensivelmente as chances de interferência. No *Bluetooth*, pode-se utilizar até 79 frequências (ou 23, dependendo do país) dentro da faixa ISM, cada uma "espaçada" da outra por intervalos de 1 MHz[4].

Os dispositivos são classificados, de acordo com a potência e alcance, em três níveis: classe 1 (100 mW, com alcance de até 100 m), classe 2 (2,5 mW e alcance até 10 m) e classe 3 (1 mW e alcance de 1 m, uma variante muito rara).

Os dispositivos *Bluetooth* comunicam-se entre si e formam uma rede denominada piconet (rede formada pela conexão entre dispositivos Bluetooth), na qual podem existir até oito dispositivos interligados, sendo um deles o mestre (*master*) e os outros dispositivos escravos (*slave*); uma rede formada por diversos "*masters*" (com um número máximo de 10) pode ser obtida para maximizar o número de conexões. Esse procedimento consiste em fazer uma piconet se comunicar com outra que está dentro do limite de alcance, esquema este denominado *scatternet*.

Existem dois modos de modulação: *Basic Rate* (BR), que é obrigatória e usa uma modulação binária FM para minimizar a complexidade de transmissão e a *Enhanced Data Rate* (EDR) que é um modo opcional e usa a modulação PKS (*Phase-Shift Keying*) e tem duas variantes: π/4-DQPSK e 8DPSK. A taxa bruta de transmissão é 1 Mbps para a "Categoria Básica", 2 Mbps para "Categoria de Dados Melhorados"

[4] Info wester. (30 de Janeiro de 2008). *Info wester - Tecnologia Bluetooth: o que é e como funciona?* Acesso em 20 de Fevereiro de 2017, disponível em https://www.infowester.com/bluetooth.php

usando π/4-DQPSK e 3 Mbps para "Categoria de Dados Melhorados" usando 8DPSK (Okuda, 2008).

3.3.2 MÓDULOS HC-05 E HC-06

Os módulos mais comuns presentes do mercado que realizam essa comunicação *Bluetooth* são o HC-05 e o HC-06, que trabalham com a tecnologia BT 3.0. A diferença básica entre os módulos é que o HC-05 pode ser configurado nos modos *Master* (mestre), *Slave* (escravo) e *Loopback*, e o HC-06 apenas no modo *Slave*.

- Modo *Master* (Mestre): O módulo pode se conectar a outros dispositivos *Bluetooth*;
- Modo *Slave* (Escravo): O módulo apenas recebe conexões de outros dispositivos *Bluetooth*;
- Modo *Loopback*: O módulo recebe os dados do módulo *Master* e envia de volta esses mesmos dados. É um modo utilizado geralmente para testes.

Tanto o HC-05 quanto o HC-06 podem ser configurados por meio de comandos AT, a partir de programas de controle serial. O HC-05 possui mais comandos, como pode ser visualizado na Tabela 2.

Tabela 2 - Comandos AT dos módulos HC-05 e HC-06.

Comando	Função	HC-05	HC-06
AT	Teste	X	X
AT+RESET	*Reset*	X	
AT+VERSION	Mostra a versão do *software*	X	X
AT+ORGL	Restaura configurações padrão	X	
AT+ADDR?	Mostra o endereço do módulo BT (Bluetooth)	X	
AT+NAME	Mostra/ altera o nome do módulo BT	X	X
AT+RNAME?	Mostra o nome do módulo BT remoto	X	
AT+ROLE	Seleciona modo *master/ slave/ loopback*	X	
AT+PSWD	Altera a senha do módulo	X	X
AT+UART	Altera a velocidade (*baud rate*)	X	X
AT+RMAAD	Remove a lista dos dispositivos pareados	X	
AT+INQ	Inicia a varredura por dispositivos BT	X	
AT+PAIR	Efetua o pareamento com BT remoto	X	
AT+LINK	Efetua a conexão com o BT remoto	X	

Mais recentemente, apareceram os HC-08 e HC-10 que trabalham com tecnologia BT 4.0 ou BLE ("*Bluetooth Low Energy*"). Os módulos BLE são os únicos que podem ser conectados a um *Iphone*, pois a *Apple* não fornece suporte ao BT 3.0.

O módulo HC-06 é um pouco mais simples de configurar que o HC-05. Resumidamente, basta conectá-lo a, por exemplo, um Arduíno, carregar o programa e enviar os comandos AT por meio da serial.

No caso do HC-05, para enviar esses comandos tem-se que utilizar um pino adicional, o pino KEY (como pode ser visualizado na Figura 9), que deve estar em estado ALTO (*HIGH*). Esse procedimento coloca o módulo em "modo AT". Alguns módulos vêm até mesmo com um pequeno botão para colocar esse pino em nível *HIGH*.

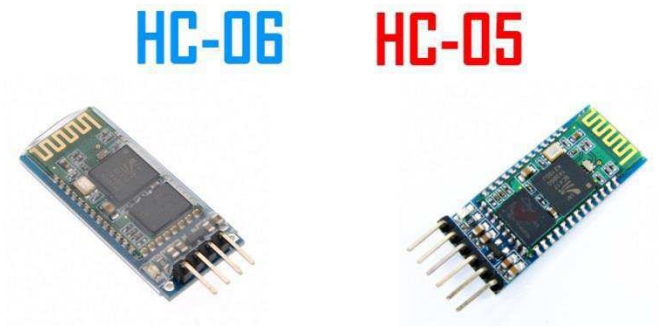


Figura 9 - Módulos Bluetooth.

A conexão dos pinos dos módulos é extremamente simples, os dois tipos possuem os pinos Vcc, GND, Rx e Tx. É importante ressaltar que o nível de sinal utilizado pela maioria dos módulos para a comunicação serial é de 3.3 V (não confundir com a alimentação do módulo, que pode ser de 3.6 a 6 V). Assim, para usar o módulo com as placas Arduíno, por exemplo, que trabalham com nível de sinal de 5 V, deve-se usar sempre um divisor de tensão para o pino RX do módulo para evitar danos ao mesmo.

Vale salientar que ao se usar a entrada serial do UNO para o uso do módulo, é muito importante lembrar que o mesmo não pode estar fisicamente conectado aos pinos 0 e 1 (Tx e Rx) do Arduíno durante a carga do programa. Isso porque o USB também usa essa mesma serial. Uma maneira simples de contornar esse problema (se o projeto não utiliza muitos GPIOs do UNO) é usar uma porta serial por SW por meio da *library SoftwareSerial*.

3.4 MÓDULO NRF24L01

O módulo NRF24L01 é fabricado pela Nordic e é uma opção de comunicação *Wirelless* que pode ser conectado ao Arduíno, PIC, Raspberry, dentre outros

microcontroladores. Este é um *transceiver* (transceptor), um dispositivo que combina um transmissor e um receptor utilizando componentes de circuito comum para ambas às funções em um só módulo.

O módulo tem capacidade de receber sinais de até seis transmissores sem que haja interferência entre os mesmos. Isso é possível devido ao conjunto de seis *Pipe*'s (Tubos) de dados com endereços únicos que o chip possui. Um *Pipe* é um canal lógico que possui um endereço físico único no canal de RF decodificado no chip nRF24L01+. Sendo assim para estabelecer uma comunicação entre um transmissor e um receptor é preciso configurar o mesmo endereço nos módulos.

Seu alcance pode chegar a 10 metros em ambientes internos e 50 metros em campo aberto. O módulo acompanha uma antena embutida que opera na frequência de 2,4 GHz com velocidade de operação que chega a 2 Mbps, modulação GFSK, habilidade de anti-interferência, verificação de erros por CRC, comunicação multiponto de 125 canais e controle de fluxo. Outras características estão listadas a seguir.

- Corrente durante a transmissão: 11,3 mA;
- Corrente durante a recepção: 13,5 mA a velocidade de 2 Mbps;
- Corrente em repouso 900 nA;
- Temperatura de trabalho: -40 a 85 ºC;
- Tensão de alimentação: 1,9 a 3,6 V (recomendado 3,3 V). Os pinos de sinal podem trabalhar normalmente com nível de sinal de 5 V.[5]

O módulo utiliza o protocolo de comunicação SPI (*Serial Peripheral Interface* - Interface Periférica Serial), abordada em tópico anterior, o que permite assim a comunicação com os vários modelos de microcontroladores. A Figura 10 apresenta a pinagem do módulo, e a Tabela 3 apresenta a função de cada pino.

Para o Arduíno (que foi o microcontrolador escolhido neste trabalho para desenvolvimento das aplicações com os módulos), os pinos de controle CSN e CE podem usar qualquer porta digital do mesmo, devendo ser identificados no programa por meio da função "RF.radio()", que faz parte da biblioteca do módulo para o Arduíno. Contudo, os pinos de comunicação SCK, MOSI, MISO tem que ser necessariamente ligados aos pinos destinados à comunicação SPI.

---

[5] Nordic Semiconductor. (s.d.). *nRF24L01+ Single Chip 2.4GHz Transceiver - Product Specification v1.0.* Acesso em 20 de fevereiro de 2017, disponível em www.buildbot.com.br: http://www.buildbot.com.br/files/nRF24L01P_Product_Specification_1_0.pdf

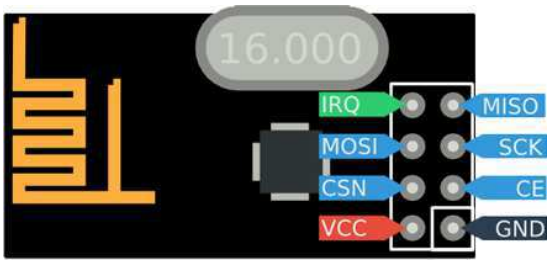


Figura 10 - Pinagem do módulo NRF24L01.

Tabela 3 - Funções dos pinos do NRF24L01.

Pino	Função	Descrição
VCC	Alimentação	+1,9V a +3,6V DC
GND	Alimentação	Terra (0 V)
CE	Entrada digital	*Chip Enable* (ativa os modos RX ou TX)
CSN	Entrada digital	*SPI Chip Select*
SCK	Entrada digital	*SPI Clock*
MOSI	Entrada digital	*SPI Slave Data Input*
MISO	Saída digital	*SPI Slave Data Output*, com opção de *tri-state*
IRQ	Saída digital	Pino de interrupção (ativo em nível baixo)

Como comentado, para a implementação do mesmo com o Arduíno, é necessário adicionar a biblioteca do módulo. Durante a pesquisa, foram encontradas duas bibliotecas, a ‘nRF24L01.h” e a "RF24.h". Suas aplicações em testes podem ser vistas no tópico 0.

3.5 MÓDULO HC – 12

O módulo *wireless* HC-12 é também um transceptor, como o NRF24L01, porém sua comunicação é através do protocolo RS232, nível TTL (UART - *Universal Asynchronous Receiver/Transmitter*). Trabalha na faixa de 433,4 a 473 MHz. Essa faixa de frequência pode ser configurada utilizando comandos AT. Além disso, é possível alterar outros parâmetros, como a potência de transmissão do módulo (máximo de 100 mW).[6] Este módulo se assemalha aos módulos *bluetooth* HC-05 e HC-06, pois trabalha com comunicação serial com o microcontrolador, ou seja, são necessários apenas dois

[6] Arduíno e CIA. (29 de Novembro de 2016). *Arduino e CIA - Comunicação sem fio com módulo wireless HC-12 e Arduino*. Acesso em 15 de Fevereiro de 2017, disponível em www.arduinoecia.com.br: http://www.arduinoecia.com.br/2016/11/modulo-wireless-hc-12-arduino.html.

pinos para ligação do RX, TX e a alimentação do módulo. Outras características estão listadas a seguir:

- Tensão: 3,3 V a 5 V;
- Corrente nominal: 16 mA;
- Potência máxima: 100 mW;
- Alcance: Média de 1000 m;
- Sensibilidade: 10 dBm a -30 dBm.[7]

Na Figura 11 podem ser observadas, a princípio, duas formas de ligar uma antena nesse módulo, já que ele não conta com antena embutida. A primeira é soldar uma antena tipo "mola" na placa (essa antena já vem com o módulo), e a outra é utilizar uma antena externa no conector U.FL.

O HC-12 também tem um pino para que o módulo entre em modo de comando AT (pino SET), como já comentado, que é ativado em nível baixo. Com isso, é possível configurar os parâmetros do módulo por meio de comandos AT.

Figura 11 - Pinagem do módulo HC-12.

Utilizou-se, para realizar a configuração do módulo, um conversor USB-TTL FTDI RS232, mas também é possível utilizar um microcontrolador, como o Arduíno, para fazer esse processo. A montagem para tal é apresentada na Figura 12, tem-se uma alimentação externa para o módulo, porém para a simples configuração dos parâmetros do módulo, a própria alimentação do conversor foi suficiente.

Observe que o pino SET do módulo está ligado ao GND, o que indica que o HC-12 deve entrar em modo de comando AT. Foi utilizado um programa para a comunicação através da porta serial, o Termite. Porém o próprio monitor serial da IDE do Arduíno cumpre esse papel.

---

[7] Elecrow. (s.d.). *HC-12 Wireless serial port comunication module - User manual.* Acesso em 10 de Janeiro de 2017, disponível em www.elecrow.com: https://www.elecrow.com/download/HC-12.pdf

Configurada a velocidade padrão do módulo, que é de 9600 bps, e testada a comunicação usando o comando AT, o módulo responde com um "OK", indicando que a comunicação está funcionando.

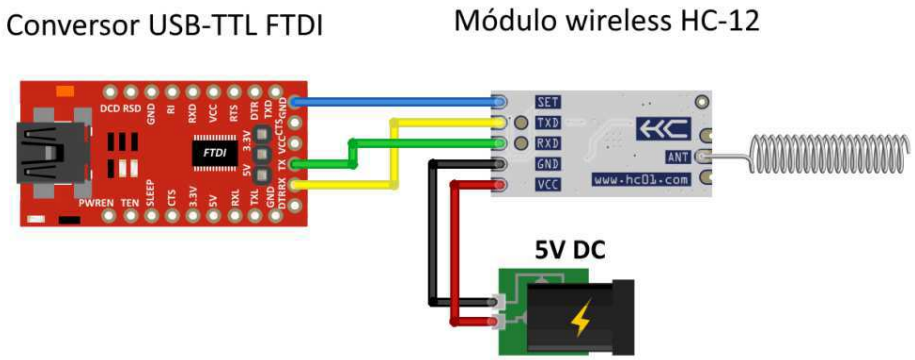


Figura 12 - Ligação para configuração do módulo HC-12.

A seguir, tem-se a lista dos comandos para as configurações do módulo:

- AT - Instrução de teste. Retorna "OK";
- AT+Bxxxx - Altera a velocidade da porta serial, onde xxxx é a velocidade desejada: 1200 bps, 2400 bps, 4800 bps, 9600 bps, 19200 bps, 38400 bps, 57600 bps e 115200 bps. A velocidade padrão do módulo é 9600 bps. Após o comando, o módulo retorna com a mensagem "OK+Bxxxx";
- AT+Cxxx - Altera o canal de comunicação *wireless*, onde xxx aceita valores entre 001 e 127. O valor padrão é 001, operando na frequência de 433.4 MHz. Após o comando, o módulo retorna com a mensagem "OK+Cxxx";
- AT+Px - Altera a potência de transmissão do módulo, onde x pode assumir valores de 1 a 8: 1 (-1 dBm), 2 (2 dBm), 3 (5 dBm), 4 (8 dBm), 5 (11 dBm), 6 (14 dBm), 7 (17 dBm) e 8 (20 dBm). Após o comando, o módulo retorna com a mensagem "OK+Px".

Após as configurações, e posterior uso do módulo, o pino SET deve ser desconectado.

3.6 MÓDULO ESP8266

ESP8266 é um chip de arquitetura 32 bits com Wi-Fi integrado, que mede apenas 5 mm x 5 mm. O núcleo da CPU é baseado em um IP Xtensa, da Cadence, que foi modificado a critérios da Espressif. Por ser tão pequeno e sua utilização se tornar

complicada, uma outra empresa chinesa, a AI-Thinker, passou a produzir módulos utilizando o chip e adicionando alguns componentes extras (Memória adicional, memória EEPROM, antena, entre outros). Algumas de suas características estão listadas a seguir:

- É um *System-On-Chip* com Wi-Fi embutido;
- Tem conectores GPIO, barramentos I2C, SPI, UART, entrada ADC, saída PWM e sensor interno de temperatura;
- CPU que opera em 80 MHz, com possibilidade de operar em 160 MHz;
- Arquitetura RISC de 32 bits;
- 32 KBytes de RAM para instruções;
- 96 KBytes de RAM para dados;
- 64 KBytes de ROM para *boot*;
- Possui uma memória Flash SPI Winbond W25Q40BVNIG de 512KBytes;

Os módulos ESP8266 são fornecidos numa ampla variedade de modelos, com diferenças perceptíveis principalmente no que tange à quantidade de IOs disponíveis para acesso externo e no tamanho do módulo. A Figura 13 apresenta alguns desses modelos.

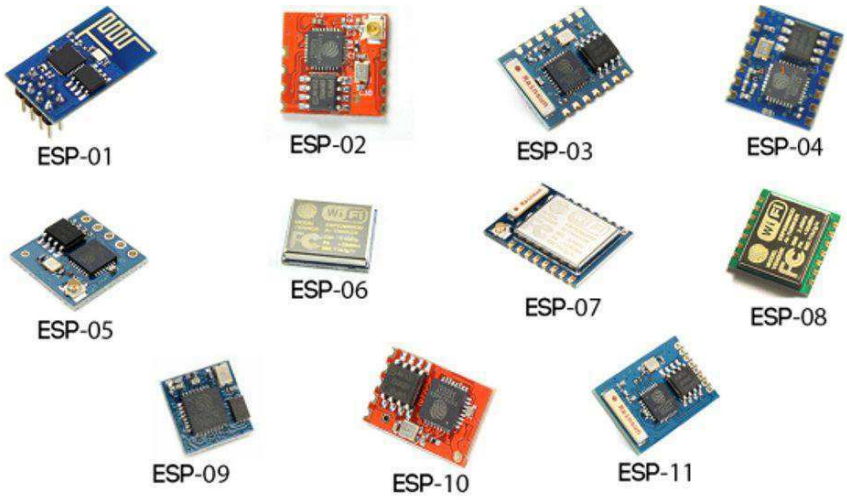


Figura 13 - Modelos de ESP.

Assim como o Arduino, o ESP8266 pode utilizar o mesmo software para sua programação, possuindo inclusive compatibilidades com as bibliotecas (Santos, Pereira, Pereira, & Neto, 2016).

Alguns dos módulos vêm pré-carregados com um *firmware* que os transformam em "Pontes Serial-WiFi". Para realizar essa ponte, a interface serial dos módulos

obedece a uma tabela de comandos. Os comandos seriais seguem o padrão AT, e variam de *firmware* para *firmware*, diferenciando principalmente no tipo de resposta dada pelo módulo a cada comando, e na quantidade de comandos suportados. Vale lembrar que o tensão de operação dos pinos do módulo é de 3,3 V.

3.6.1 PLACA WEMOS

A WeMos (Figura 14), à primeira vista, assemelha-se a um Arduíno Uno, porém com o ESP8266 no lugar do Atmega (168 ou 328)[8]. Algumas de suas características estão listadas a seguir:

- Microcontrolador: ESP-8266EX;
- Tensão de Operação: 3.3 V;
- Entradas Digitais: 11;
- Entradas Analógicas: 1 (Máx. *input*: 3.2 V);
- *Clock Speed*: 80 MHz/160 MHz;
- Memória: 4M bytes;
- Comprimento: 68.6 mm;
- Largura: 53.4 mm;

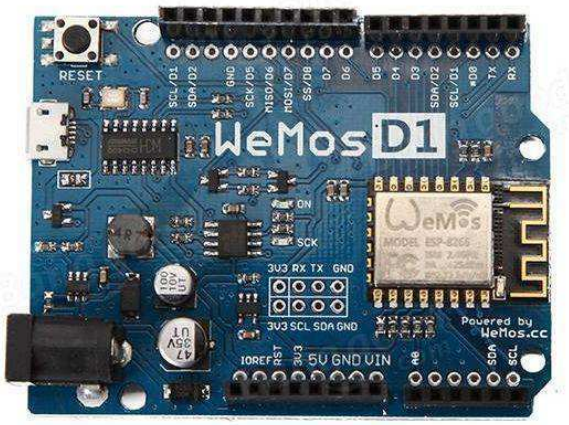


Figura 14 - Placa Wemos.

A WeMos é suportada pela IDE Arduíno. Para a instalação, os passos estão descritos no Apêndice B.

[8] Wemos Electronics. (s.d.). *Wemos Electronics*. Acesso em 07 de Março de 2017, disponível em www.wemos.cc: https://www.wemos.cc/tutorial

Assim como o Arduíno Pro-mini, já existe no mercado a versão mini da Wemos, que traz as mesmas características, mas com o tamanho bastante reduzido. A Figura 15 apresenta o mapa de pinos desta placa.

Assim como todas as placas ESP, a Wemos adentra nas aplicações da Internet das Coisas (IoT), que pode ser dita como uma revolução tecnológica que tem por objetivo conectar dispositivos á Internet, para que seu acionamento ou leitura, no caso de sensores, possa ser realizado em qualquer lugar do mundo, diminuindo espaços e interpretando informações.

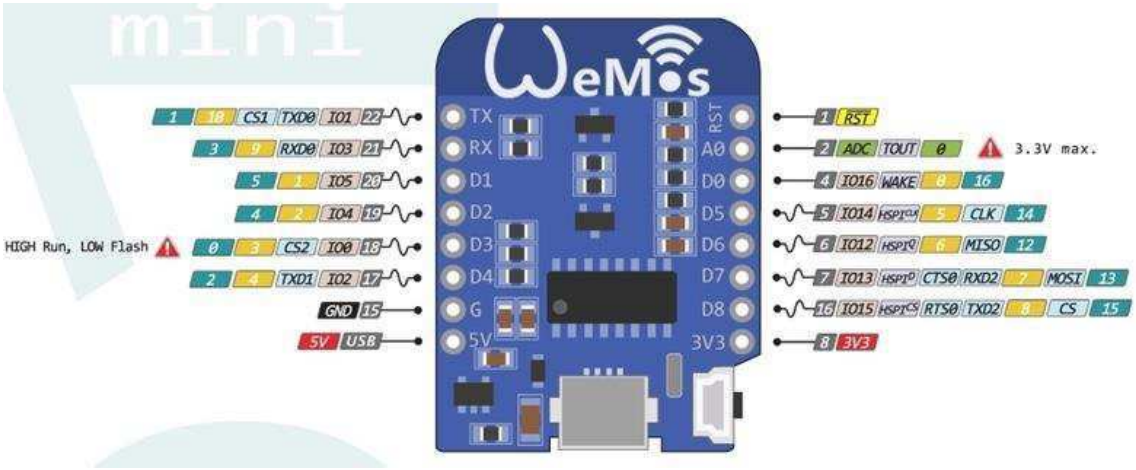


Figura 15 - Mapa de pinos do Wemos Mini.

3.7 COMPARAÇÃO ENTRE OS MÓDULOS

A Tabela 4 apresenta um resumo das características dos módulos.

Tabela 4 - Comparação entre os módulos de comunicação.

Módulo	Tecnologia de comunicação	Modulação	Frequência de operação	Tensão de alimentação (V)	Potência	Alcance (m)	Taxa máx. transmissão de dados (Kbps)
RF 433 MHz	RF	AM	433 Mhz	3,5 a 12	10 mW	20 a 200	32
HC - 06	Bluetooth	FM ou PSK	2,45 GHz	3,6 a 6	1 – 2,5 - 100 mW	1, 10 ou 100	1382,4
NRF24L01	SPI	GFSK	2,4 GHz	1,9 a 3,6	7 dBm	Até 50	250
HC – 12	RS - 232	-	433.4 a 473 MHz	3.3 V à 5 V	100 mW	Até 1000	115,2
ESP 8266	Wi-Fi	-	2,4 GHz	3,3	20 dBm	-	54000

Como já mencionado anteriormente, é de extrema importânica uma escolha bem orientada dos componentes a serem utilizados em determinada aplicação. Todos os módulos apresentados neste trabalho tiveram suas finalidades comprovadas a partir de testes que estão descritos nas seções seguintes Porém, cada um possui características distintas, para determinadas funcionalidades uns podem não servir, ou podem ser mais eficazes que outros.

4 COMPONENTES ELETRÔNICOS E ELETRO-MECÂNICOS

Neste tópico são apresentados componentes e circuitos que foram utilizados para os testes e aplicações dos módulos descritos anteriormente.

4.1 RELÉS

Relés são as chaves de comando que operam com sinais elétricos de baixa potência e controlam uma grande maioria dos circuitos. Um exemplo de uma aplicação de relé ideal é configurar um sinal elétrico para controlar vários circuitos, permitindo o completo isolamento da eletricidade entre o controlador e os circuitos controlados.[9].

Um relé atua como um isolante que protege o dispositivo que está sendo usado. Quando o controle (extremidade de entrada) ou a carga (extremidade de saída), não estão eletricamente conectados, o relé impede quaisquer danos por surtos de energia na sua aplicação.

A conexão de uma carga com o módulo relé (seja eletromecânico ou SSR), é basicamente a apresentada na Figura 16. Alguns módulos relé possuem três pinos do lado DC: Vcc, GND e SINAL , e três contatos do lado AC que configuram as conexões NA e NF. No exemplo, está sendo realizado o acionamento de uma lâmpada por meio

[9] Omega. (2015). *Omega - Sua fonte para medição e controle de processos.* Fonte: http://br.omega.com/artigos-tecnicos/diferentes-tipos-de-reles-mecanicos-ssr-internos-ou-externos.html

de um relé que possui apenas dois contatos do lado DC, nesse caso é conectado diretamente o pino de sinal e GND. A conexão do sinal que provém de uma saída digital do Arduíno. Do lado AC, a lâmpada é conectada em série com a fonte AC e o contato NA do relé. Assim, quando o pino SINAL receber um nível alto (5 V), o contato NA do relé irá fechar, ligando a lâmpada.

Este contato do SINAL poderia ser proveniente também, de qualquer um dos módulos mencionados. A montagem foi realizada em laboratório, inclusive utilizando os módulos de comunicação para a atuação do pino SINAL.

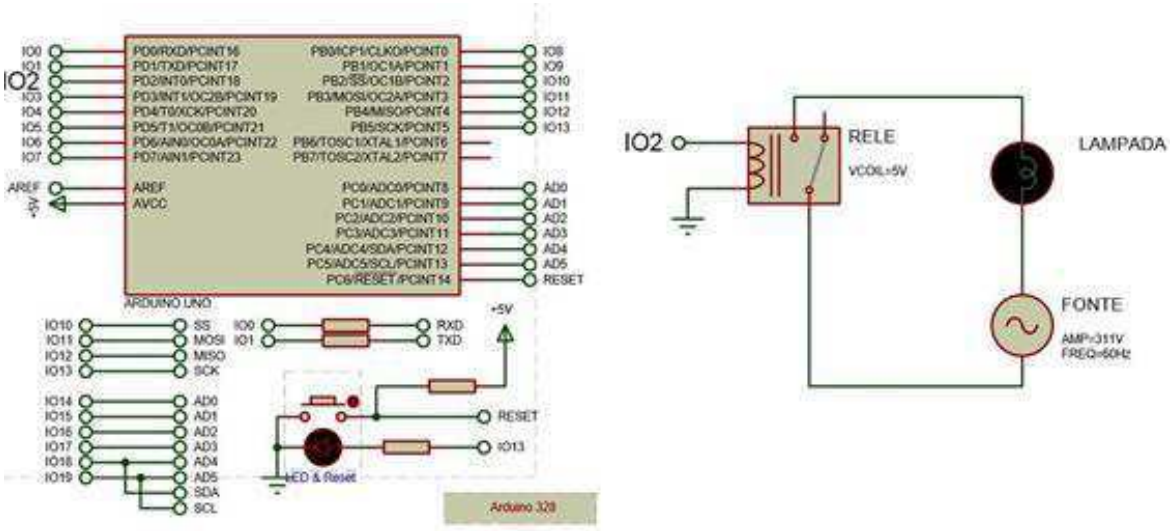


Figura 16 - Acionamento de uma lâmpada por relé.

4.1.1 RELÉ ELETROMECÂNICO

Os relés eletromecânicos são caracterizados pelo uso de uma bobina eletromagnética energizada por uma corrente para abrir ou fechar um circuito. O campo magnético puxa ou empurra o braço do interruptor, removendo ou iniciando o contato (Figura 17).

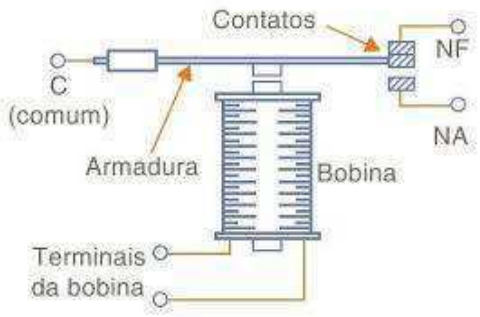


Figura 17 - Relé eletromecânico.

4.1.2 SSR

O SSR (*Solid State Relay* – Relé de estado sólido) cumpre a mesma função de um relé eletro-mecânico, porém não possui peças móveis. Eles são dispositivos semicondutores que usam luz, ao invés de magnetismo, para acionar seus contatos. A maioria dos sistemas que utilizam relés de estado sólido, ou SSR, tem um diodo emissor de luz. A extremidade de carga do espaço aberto detecta a luz, acionando um interruptor de estado sólido que controla a abertura e fechamento do circuito. A Figura 18 a seguir apresenta um exemplo desse tipo de relé.

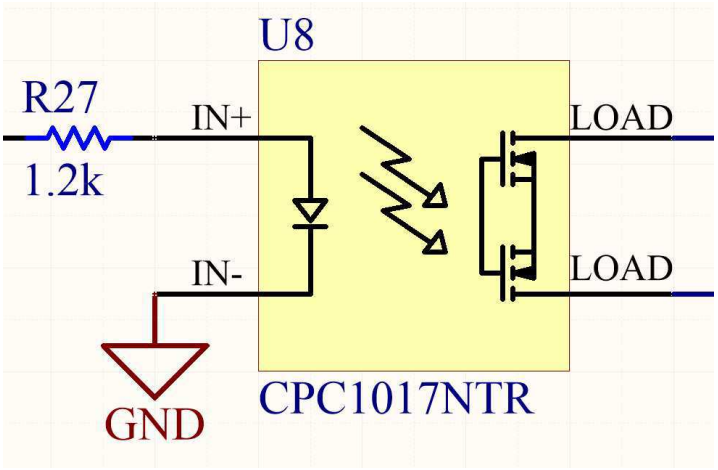


Figura 18- Relé de estado sólido.

A vantagem de usar relés de estado sólido é que eles protegem circuitos contra ruídos elétricos. SSRs não têm vibração nos contatos, têm baixa IEM/RFI (interferência eletromagnética/de rádio frequência) e, comparados com relés mecânicos, têm uma vida mais longa, por não apresentarem peças móveis. Entretanto, um relé de estado sólido pode ser utilizado apenas com disjuntores monopolares.

4.2 CIRCUITO COM ACOPLAMENTO ÓPTICO

Para o acionamento de cargas com os sinais oriundos dos módulos de comunicação, também foi realizada a conexão por acoplamento óptico. Os acopladores ópticos funcionam por meio de um feixe de luz, para transmitir sinais de um circuito para outro, sem a ligação física entre eles, fazendo assim o isolamento, protegendo um circuito do outro. O circuito de conexão da carga, com este pode ser observado na Figura 19.

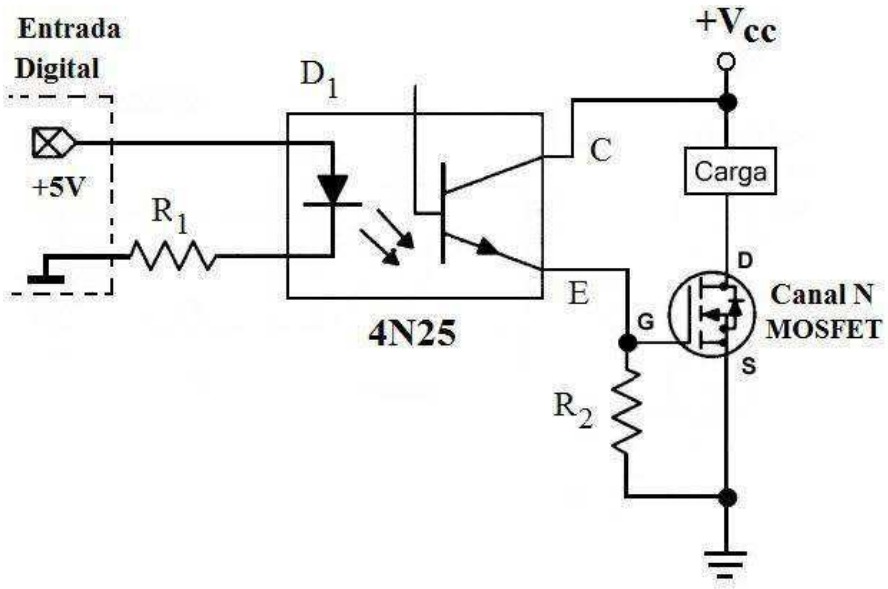


Figura 19 - Circuito para acionamento de carga por acoplamento óptico.

Existem diversos CIs que cumprem essa função, o utilizado na montagem foi o 4N25 (seu diagrama de pinos pode ser visualizado na Figura 20), e o mosfet utilizado foi o IRF540. Para fins de verificação do funcionamento do circuito, foi utilizado como carga um motor DC de 5V, a partir desse circuito é possível acionar qualquer carga, modificando apenas a alimentação.

Assim, quando um sinal é inserido na entrada do acoplador, irá fluir corrente do coletor para o emissor do transistor do acoplador, alimentando o *gate* do mosfet que também passará a conduzir, acionando, a partir de então, a carga.

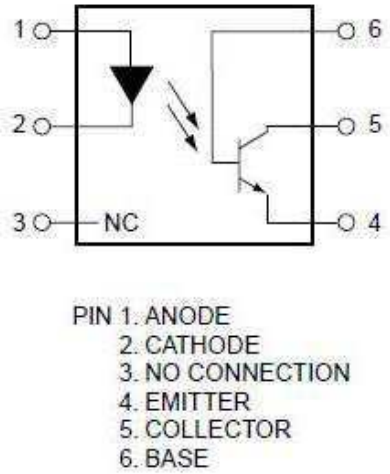


Figura 20 - Diagrama de pinos do 4N25.[10]

[10] Motorola - SEMICONDUCTOR TECHNICAL DATA. (s.d.). *ALLDATASHEET.COM.* Acesso em 20 de Março de 2017, disponível em http://pdf1.alldatasheet.com/datasheet-pdf/view/2846/MOTOROLA/4N25.html

4.3 SERVO MOTORES

O servo-motor (Figura 21) é um componente eletromecânico que recebe sinais elétricos, o transforma em um movimento giratório de seu eixo, sendo esse movimento proporcional ao sinal recebido. As suas características mais importantes são o seu torque (força rotacional), velocidade de operação, peso, tamanho e consumo. Geralmente essas informações são fornecidas pelo fabricante.

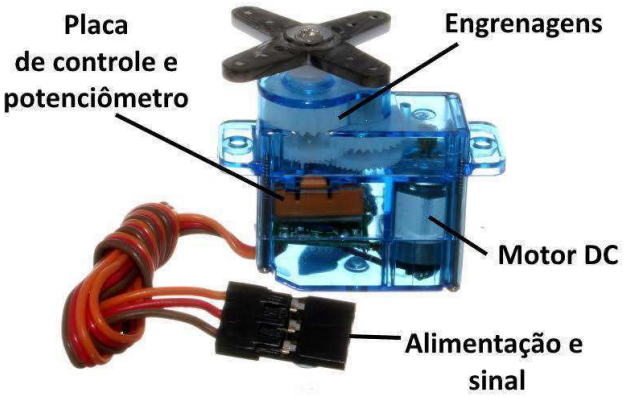


Figura 21 - Servo-motor.

Eles possuem três terminais: terra (GND), alimentação (Vcc) e o terceiro terminal é destinado ao pino de controle. Este pino de controle define o ângulo de rotação do motor dos servos, e é determinado pela duração do pulso que se aplica nessa entrada. O servo funciona recebendo um sinal no formato PWM (*Pulse Width Modulation*, modulação por largura de pulso), sistema que consiste em gerar uma onda quadrada em que se varia a duração do pulso, mantendo o período da onda. Este sinal é de 0 volts ou 5 volts. O circuito de controle do servo monitora este sinal em intervalos de 20 ms, se dentro deste intervalo ele perceber uma alteração do sinal de 0 V para 5 V durante 1 ms até 2 ms, ele modifica a posição do seu eixo para coincidir com o sinal que recebeu (Figura 22).

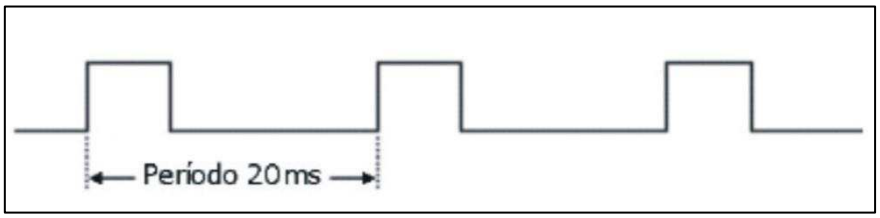


Figura 22 – Período de onda para monitoramento do PWM para um servo.

A largura mínima e máxima do pulso (sinal recebido) depende do tipo de servo. No entanto, e no caso geral, se o servo receber na sua entrada pulso com a duração de 1.5 ms, o seu eixo roda até ficar estável no centro do intervalo de rotação, a que corresponde o ângulo de 90º. Se receber pulso com a duração de 1ms, roda, no sentido anti-horário, até atingir o limite do intervalo de rotação correspondente a 0º. Se receber pulso com a duração de 2 ms, roda, no sentido horário, até atingir o outro limite do intervalo de rotação correspondente a 180º ou um pouco mais. Pulsos ente 1ms e 1.5 ms farão com que o servo rode para posições intermédias entre 0º e 90º, enquanto pulsos entre 1.5ms e 2ms farão com que o servo rode para posições intermédias entre 90º e 180º (Figura 23). Se um mesmo pulso for aplicado ciclicamente na entrada do servo ele mantém-se na mesma posição angular. Sabe-se que a duração dos impulsos e os valores angulares de rotação do eixo dependem dos fabricantes. Os valores indicados acima são valores médios.

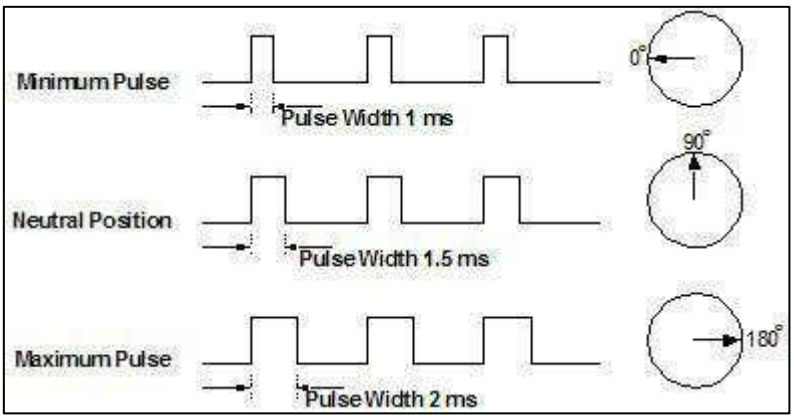


Figura 23 – PWM para controle do servo.

Para as montagens realizadas utilizando os servos, quando o mesmo possuía uma característica de torque mais elevada, não era possível sua alimentação por meio do Arduíno, pois a corrente deste não era suficiente para o acionamento do servo, então, foi necessária a utilização de uma fonte externa, como apresentado na Figura 24 a seguir.

Os canais de alimentação do servo (Vcc e GND, fios vermelho e preto), são conectados diretamente na bateria, e o pino de controle (fio amarelo) no Arduíno. É importante lembrar que os terras da placa e da bateria devem estar conectados, para que todo o circuito tenha a mesma referência.

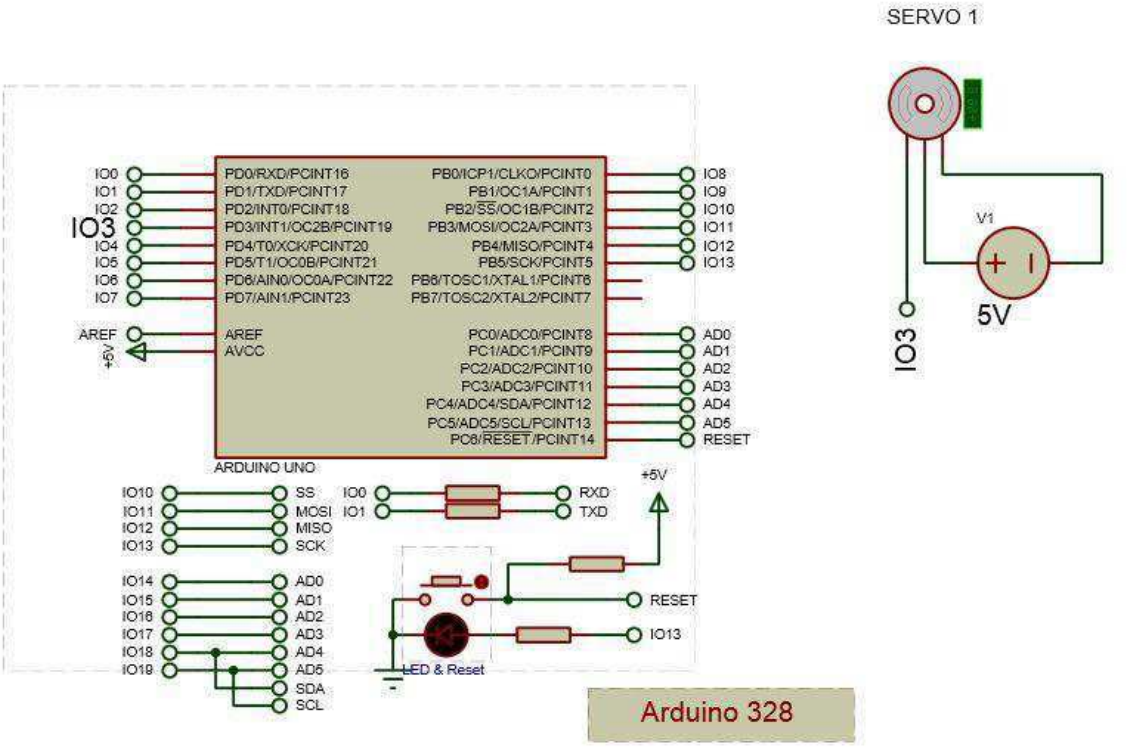


Figura 24 - Conexão do servo com o Arduíno.

4.4 PLACA FTDI – CONVERSOR USB/ SERIAL (CH-340)

O padrão USB foi desenvolvido com a ideia de facilitar a instalação e uso de periféricos nos equipamentos eletrônicos. Os cabos utilizados na comunicação USB possuem quatro fios, dois desses são reservados para a alimentação e os outros dois à comunicação.

NO protocolo USB as mensagens são enviadas sob a forma de pacotes, cada pacote é composto por setores e cada setor transporta uma informação bem definida. Isto porque cada pacote possui um setor de endereço destinado à identificação do dispositivo que receberá a mensagem.

Em uma comunicação USB, o computador associado a essa mesma comunicação é denominado *HOST*. Esse computador possui um *software* que efetua o controle do barramento do sistema em comunicação. Todas as mensagens enviadas pelo barramento USB necessitam essencialmente de três pacotes. Inicialmente o *HOST* envia um pacote, *Token*, com o endereço do dispositivo com o qual será realizada a comunicação. Este pacote é essencial à comunicação, pois existe a possibilidade de vários dispositivos estarem conectados às portas USB do computador. Em seguida o dispositivo ou o

HOST envia um pacote com os dados a transmitir. A comunicação é finalizada com o envio de um pacote de reconhecimento, garantindo que o receptor recebeu a mensagem. Pode eventualmente existir um quarto pacote utilizado para funções adicionais (Azevedo, 2010).

O conversor FTDI, é um conversor USB para serial que permite a comunicação entre o computador e outros equipamentos eletrônicos que possuam porta de comunicação no padrão TTL. FTDI são as iniciais de *Future Technology Devices International*, uma empresa especializada em tecnologia USB e que é a fabricante do chip FT232RL, cérebro da placa.

Os três principais conversores existentes no mercado são: o CH-340 (Chinês), o CP2102 (Silabs), que existe nas versões de 5 e 6 pinos, o último tendo um pino chamado DTR que tem a função de *reset*, e o PL2303 (Prolific). Cada um possui drivers específicos para instalação no sistema operacional.

O conversor utilizado neste trabalho, tanto para upload dos códigos para o Arduíno Pro-mini, como para as comunicações seriais do módulo HC-12, foi o CH-340, apresentado naFigura 25. Seus pinos de ligação são apenas TX, RX e alimentação (que pode ser de 3,3 V ou 5 V). O drive para sua instalação pode ser encontrado para *download* no seguinte site: http://www.arduined.eu/ch340-windows-8-driver-download/.

Figura 25 - Conversor CH-340.

4.5 JOYSTICK

Joysticks são formados internamente por um sistema de dois eixos ortogonais acoplados a dois potenciômetros. Esses potenciômetros executam a medição da posição da alavanca em ambos os eixos. Além disso, ele possui em um micro-interruptor, ou botão. A Figura 26 apresenta esse equipamento.

Figura 26 - Joystick.

Portanto, joysticks fornecem um sinal analógico para a posição de cada eixo (x e y), além de um sinal digital de controle para detectar o estado do botão. A ligação de seus pinos é apresentada na Figura 27. Os pinos VRx e VRy podem ser conectados em qualquer porta analógica de um microcontrolador e o pino SW em uma porta digital.

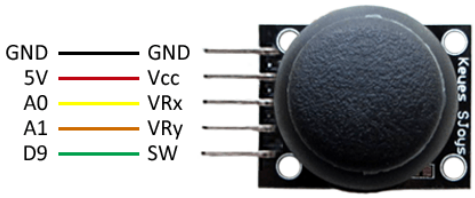


Figura 27 - Conexão dos pinos de um joystick.

5 TESTES E APLICAÇÕES COM OS MÓDULOS

5.1 TESTES COM O MÓDULO HC-06

Como teste do funcionamento do módulo *bluetooth* HC-06, foi realizada a montagem da Figura 28, onde é feita a ligação de 3 LEDs nas portas digitais 3, 4 e 5 do Arduíno. O pino RX do módulo é conectado no pino TX do Arduíno (com um divisor de tensão, garantindo no máximo 3,3 V na entrada RX do módulo) e o pino TX do módulo no RX do Arduíno. O código do programa é apresentado no Apêndice C. A comunicação com o módulo *bluetooth* foi realizada por meio de um aplicativo de celular chamado *S2 Terminal for Bluetooth*. Se enviado 1, 2 ou 3 através do aplicativo, são ligados os led's conectados as portas 3, 4 e 5, respectivamente. Se enviado A, B ou C os led's são desligados.

Com este simples teste já poderiam ser realizadas diversas aplicações de automação. Por exemplo, ao invés de leds, poderiam ser conectados relés para

acionamento de qualquer carga (de uma lâmpada, a equipamentos eletrônicos e motores DC).

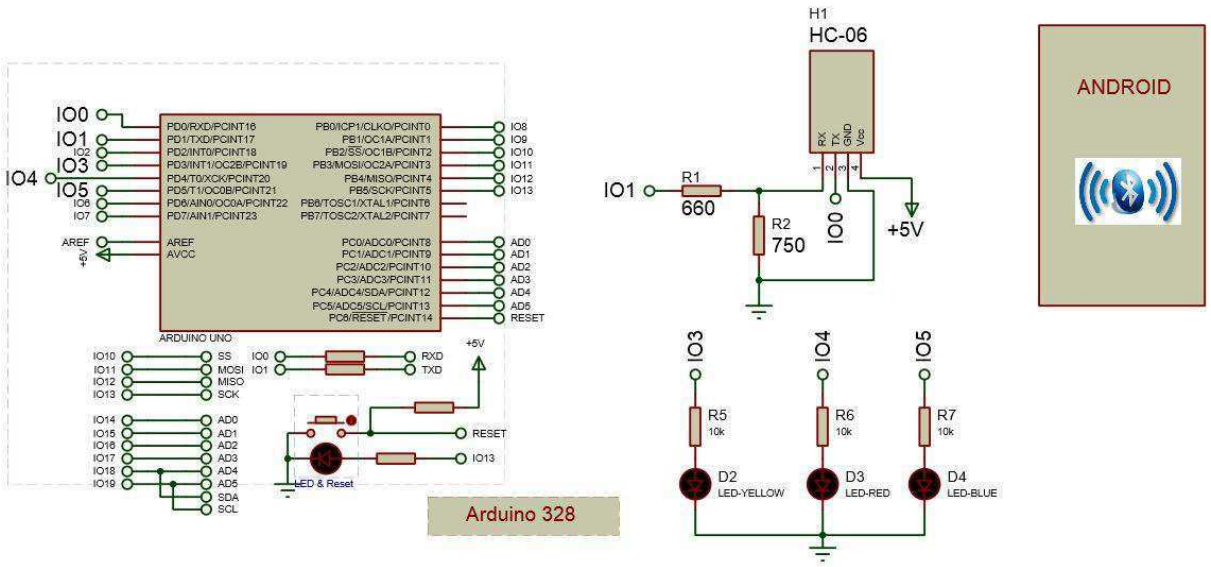


Figura 28 - Montagem para acionamento de LEDs com o módulo HC-06.

Também foi feito o teste ligando três servo-motores ao invés dos led's (Figura 29). O código está apresentado no Apêndice D. Neste caso, foi criado um sinal PWM para controlar o servo-motor, de acordo com o sinal de entrada. O sinal PWM pode ser utilizado para controlar cargas com outras aplicações que não sejam apenas o liga e desliga das mesmas: poderia ser utilizado para controlar a velocidade de um motor DC, por exemplo, pois seria aplicado o valor médio do PWM, e variando o *duty-cycle* do mesmo, ocorre a variação deste valor.

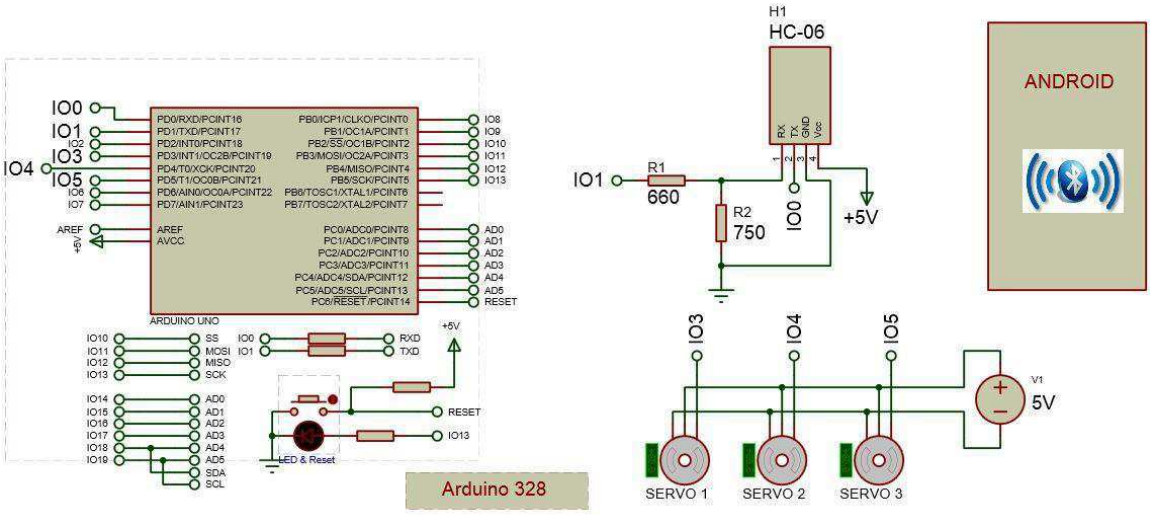


Figura 29 - Montagem para acionamento de servos-motores com o módulo HC-06.

5.2 TESTE COM O MÓDULO NRF24L01

Para fins também de teste de funcionamento do módulo, foram realizadas algumas montagens. A primeira
algumas montagens. A primeira realizada pode ser visualizada

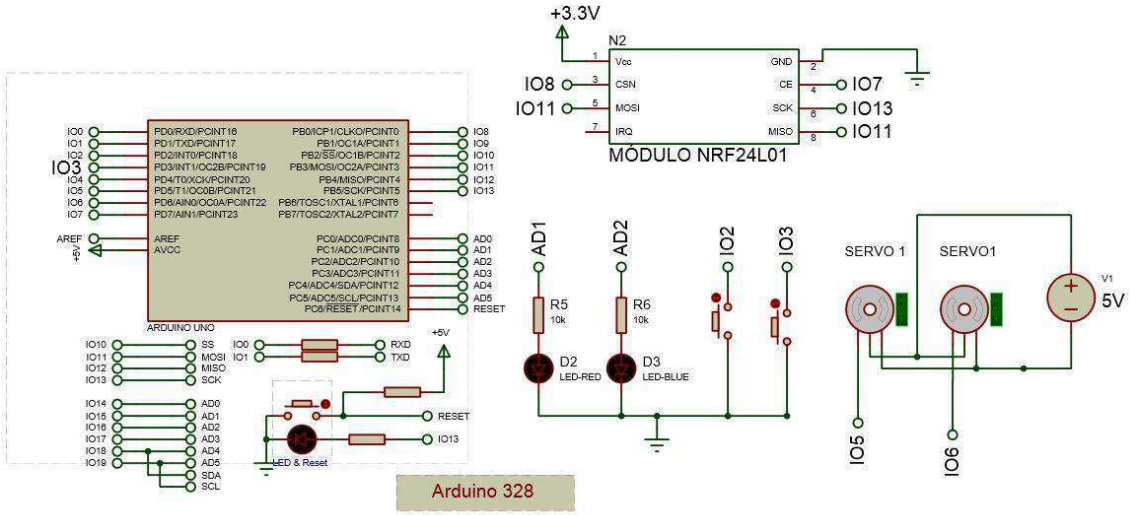
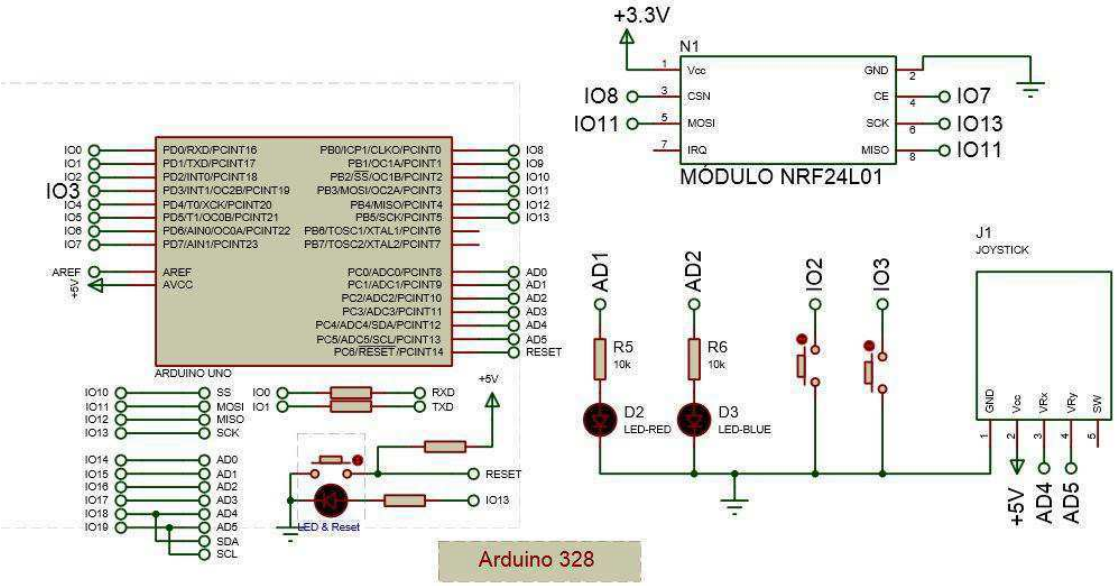


Figura 30. Nos dois lados tem-se dois leds e dois botões que acionarão esses leds nos dois sentidos da comunicação. Do lado esquerdo tem-se um joystick conectado ao Arduíno, que através do módulo controlará os dois servos que estão conectados ao Arduíno do lado direito.



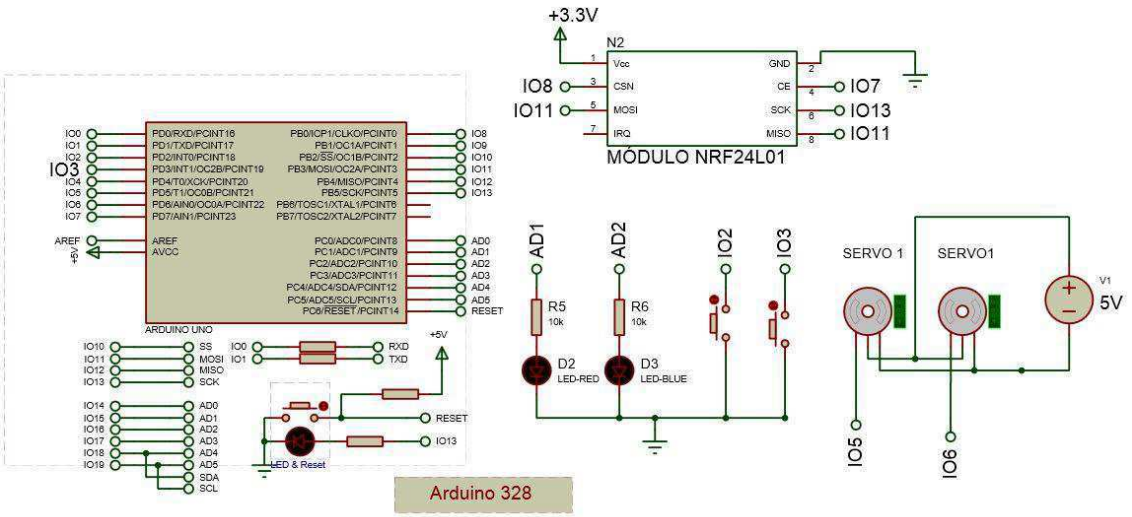


Figura 30 - Liga servos e led e servos com o NRF24L01.

Para o Arduíno UNO e suas versões compatíveis (assim como o Pró-Mini), a ligação dos pinos do módulo no mesmo é feita da seguinte forma:

- Vcc: + 3.3 V;
- GND: GND;
- CE : qualquer pino digital;
- CSN: qualquer pino digital;
- SCK: pino 13;
- MOSI: pino 11;
- MISO: pino 12;
- IRQ: SEM USO.

O código dessa montagem está apresentado no Apêndice E. Este código utilizou a biblioteca “RF24.h”, e como não utilizou a biblioteca “SPI.h”, as configurações tornaram-no mais extenso. Ele serviu para os dois módulos que se comunicavam, para tal é necessário apenas modificar o valor do objeto “radioID” para ‘0’ ou ‘1’.

A outra montagem realizada utilizou no código a biblioteca “SPI” e também a “nRF24L01.h", estas simplificaram o código. A montagem realizada está apresentada nas Figura 31 e Figura 32, e o código no Apêndice F. A ideia é acionar dois relés (que podem está conectados a qualquer carga, de acordo com suas especificações), por meio de dois botões que estão do outro lado da comunicação.

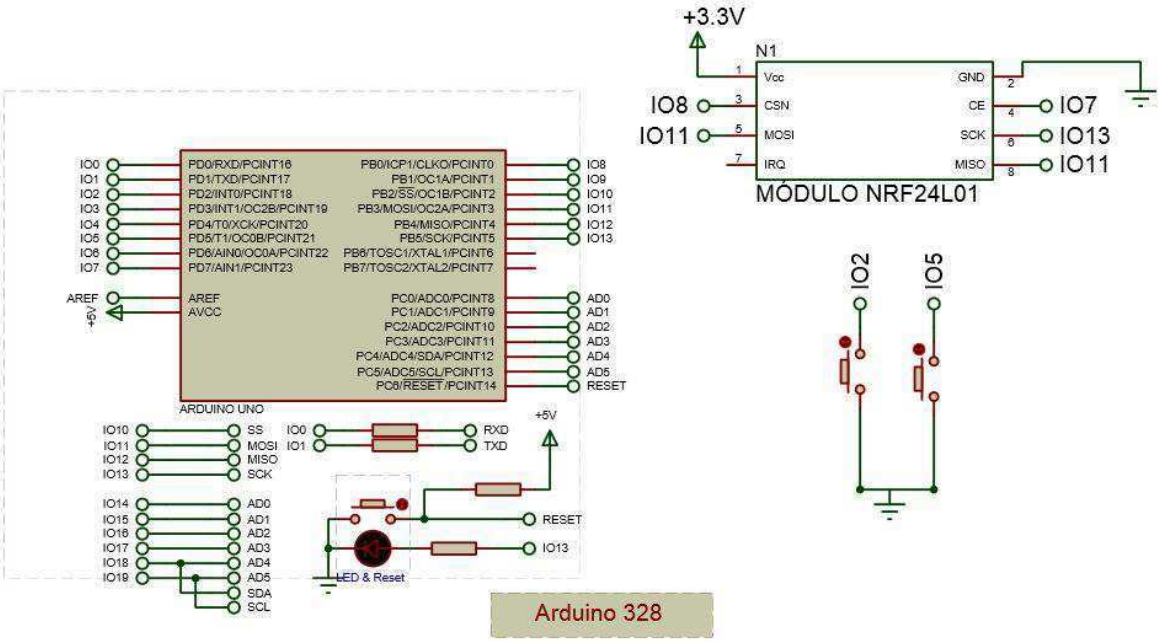


Figura 31 - Ligação Arduíno transmissor - NRF24L01.

Para a leitura do acionamento do botão em todos os testes, conectando-os a duas portas digitais do Arduíno do módulo transmissor, foi necessária a utilização da função INPUT_PULLUP no código para instanciação da porta digital como entrada.

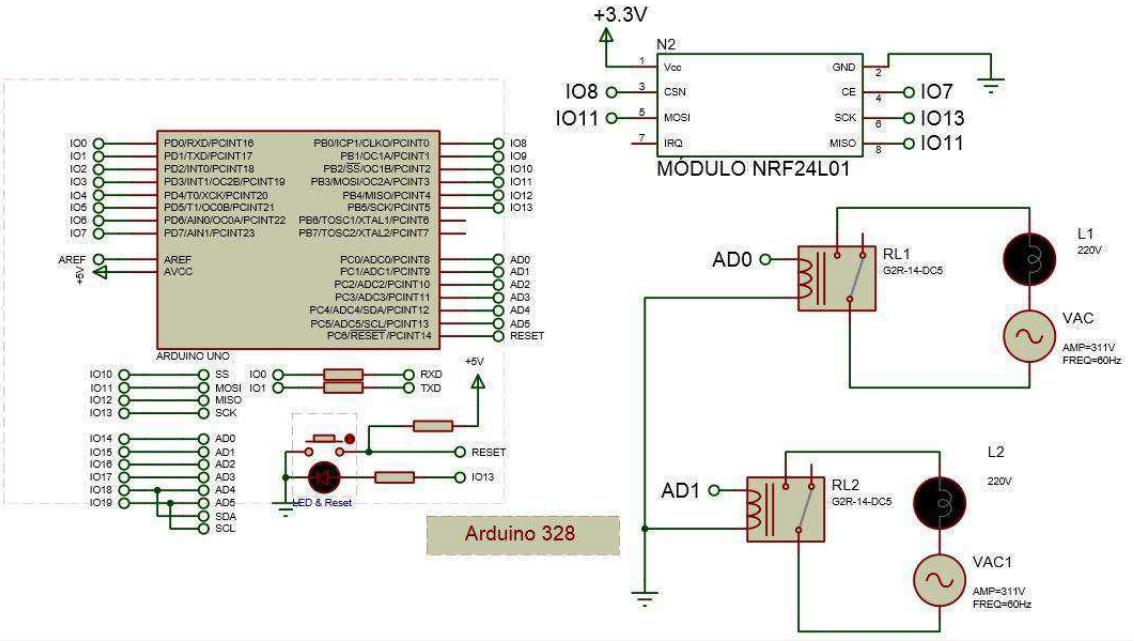


Figura 32 - Ligação Arduíno receptor - NRF24L01.

Por padrão os pinos no Arduíno estão configurados como entradas digitais, porém, para ficar mais explícito na programação, deve-se configurar o pino como

entrada. Dessa forma o pino é colocado em um estado de alta impedância, equivalente a um resistor de 100 MΩ em série com o circuito a ser monitorado. Devido a essa característica de alta impedância, quando um pino, colocado como entrada digital, encontra-se flutuando (sem ligação definida), o nível de tensão presente nesse pino fica variando, não podendo ser determinado um valor estável devido a ruído elétrico e até mesmo capacitância de entrada do pino. Para resolver esse problema é necessário colocar um resistor de *pull-up* (ligado a +5V) ou um resistor de *pull-down* (ligado a GND) conforme a necessidade. Esses resistores garantem nível lógico estável quando, por exemplo, uma tecla não está pressionada. Geralmente utiliza-se um resistor de 10 KΩ para esse propósito. A seguir é exibida na Figura 33 a ligação desses resistores no circuito para leitura de botões.[11]

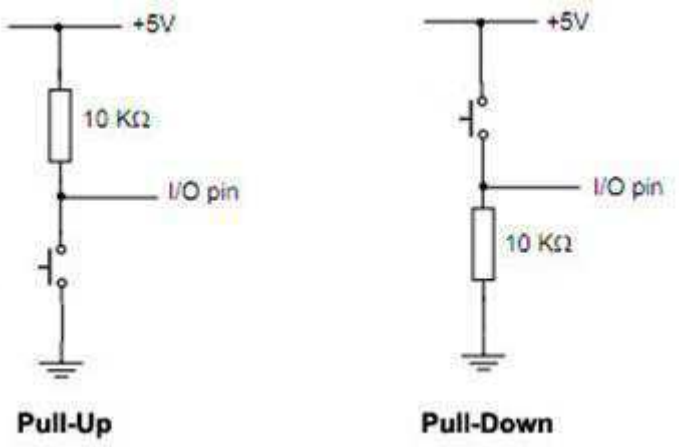


Figura 33 - Ligação dos resistores Pull-up e Pull-down.[10]

O microcontrolador ATmega328, possui resistores de *pull-up* internos, de resitência 20 KΩ, que facilitam a ligação de botões e/ou sensores sem a necessidade de conectar externamente um resistor de *pull-up*. A habilitação desses resistores é feita de maneira simples via *software*, como discorrido acima, com a descrição do pino como PULLUP.

Retomando as aplicações do módulo, do mesmo modo que o HC-06, pode-se aplicar o NRF24L01 para acionamento de cargas distintas, e ainda envio de sinal PWM para controlar o acionamentos destas cargas.

[11] Souza, F. (09 de Outubro de 2013). *Embarcados.* Acesso em 05 de Março de 2017, disponível em www.embarcados.com.br: https://www.embarcados.com.br/arduino-entradasaidas-digitais/

5.3 TESTE COM O MÓDULO HC-12

A primeira montagem realizada está apresentada na Figura 34. O módulo receberá as informações e o Arduíno irá interpretá-las para ligar ou desligar os leds. Para esse teste, a informação foi enviada através do programa de comunicação serial em um computador, onde um outro módulo HC-12 estava conectado.

O código se encontra no Apêndice G. Quando o caractere ‘1’ é enviado, o led vermelho acende, se for enviado novamente, o mesmo apaga. E assim para os outros dois com os caracteres ‘2’ e ‘3’. Esta mesma montagem também foi testada com os dois módulos, que ao invés dos dados enviados por um computador, eram botões que ligavam e desligavam os leds, onde o módulo continuou se mostrando válido.

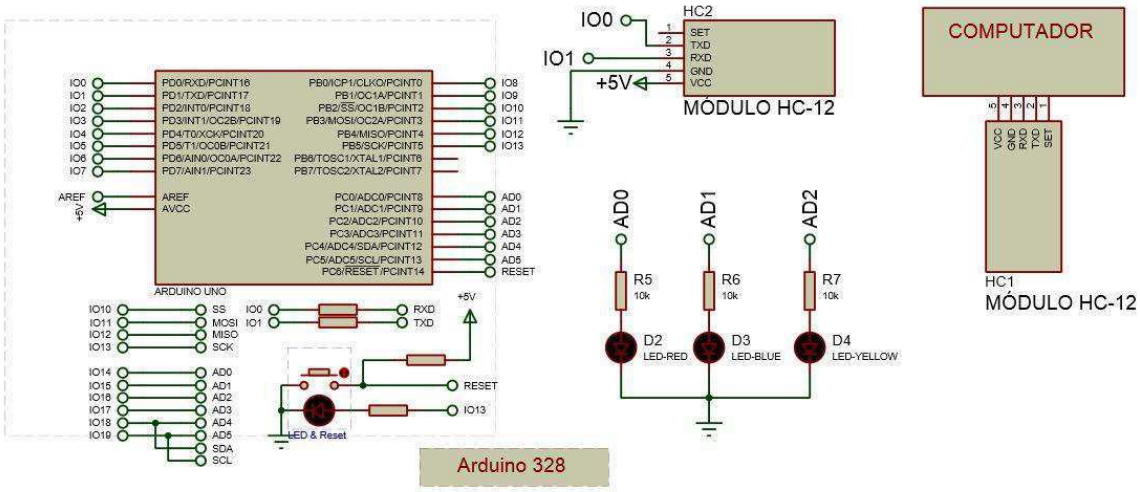


Figura 34 - Liga leds com o HC-12.

Outra montagem realizada utilizou os dois módulos conectados a um Arduíno cada, onde foi possível testar o envio da informação de sinais analógicos. O módulo transmissor converte para digital o valor da leitura analógica de um potenciômetro e o envia, e no receptor esse sinal é convertido em um sinal PWM por meio do Arduíno, controlando assim o brilho de um led. O código encontra-se no Apêndice H. Vale salientar que no código do microcontrolador receptor, a variável lida da serial deve ser instanciada no tipo ‘*byte*’, pois só assim o mesmo reconhece o valor analógico.

Este simples teste potencializa a quantidade de aplicações em que o mesmo pode ser utilizado. Também foi testado seu alcance, que se mostrou muito eficiente, inclusive em ambientes fechadas.

5.4 TESTE COM A PLACA WEMOS

Para este teste, foi realizada a ação de acender e apagar um LED através de uma página web criada a partir da programação com a placa Wemos. Na placa foi conectado um LED, em série com um resistor de 10 KΩ, na porta digital 5 do mesmo. Com o uso da biblioteca <ESP8266WiFi.h> foi possível criar um *WebServer* pelo próprio ESP.

O código para apenas a conexão da placa com a Wi-Fi é apresentado no Apêndice I. A partir deste, o módulo se conecta com a rede a qual foi inserida no código, e apresenta um IP local, onde pode ser realizada a programação para a página *web*.

O Apêndice J apresenta o código com a programação em HTML para a ação de controle do LED. A Figura 35 a seguir exibe a página criada. Percebe-se que com esse teste é possível o envio e a recepção de dados pela placa.

É importante destacar que foi utilizado um IP local. Para que a página seja acessada por meio de outra rede, deve ser utilizada uma porta aberta do roteador.

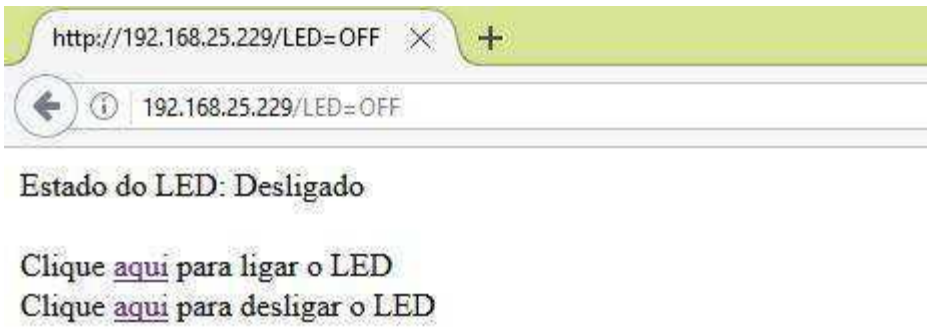


Figura 35 - Página WEB desenvolvida por meio da placa Wemos.

A partir dessas aplicações, é relevante discorrer sobre questões de segurança da informação. Existem riscos potenciais de segurança com as comunicações sem fio, uma vez que um invasor não precisa de acesso físico à rede com fio tradicional para acessar os dados (Leite, 2007).

6 CONCLUSÃO

Com este trabalho foi possível comprovar o potencial dos módulos de comunicação a distância, e a enorme gama de aplicações que os mesmos permitem desenvolver. Quando esses módulos são conectados a componentes e circuitos eletrônicos, como apresentado, eles podem acionar cargas, motores, controlar velocidades em processos, fazer monitoramento de sistemas a distância, sejam em um mesmo prédio ou sistemas que estejam do outro lado do mundo.

Em todos os testes os módulos se mostraram eficientes quanto na transmissão quanto na recepção de informações. Mesmo em testes com os obstáculos físicos (paredes comuns de tijolos entre salas), a comunicação foi garantida sem perdas. Todos os testes foram realizados no Laboratório de Eletrônica Analógica do Departamento de Engenharia Elétrica da UFCG.

Ratifica-se também, a importância de estudos aprofundados de todos os componentes que forem necessários para uma determinada aplicação, visto que cada componente possui especificações que podem fazer a diferença não só no funcionamento do equipamento, mas também na eficiência do mesmo em relação a outros que executem as mesmas funções.

BIBLIOGRAFIA

Assis, P. D. (Dezembro de 2004). MICROCONTROLADOR, Trabalho de conclusão de curso da UNIPAC - UNIVERSIDADE PRESIDENTE ANTÔNIO CARLOS. Barbacena, Minas Gerais, Brasil.

Azevedo, C. M. (Setembro de 2010). Comando e Monitorização de Sistemas de Actuação Via Rede Wireless - ZigBee. *Dissertação de Mestrado da Faculdade de Engenharia - Universidade do Porto*.

Canzian, E. (2009). Comunicação Serial – RS232. Cotia, São Paulo, Brasil.

Cunha, L. (2009). Relés e Contatores. *Setor Elétrico*, 54 - 60.

Ferreira, I. A. (Março de 2008). Sistemas de controle e supervisão de sistemas embebidos - Tipo SCADA. *Dissertação de Mestrado da Faculdade de Engenharia - Universidade do Porto*.

Leite, S. G. (Março de 2007). Comunicação Wireless - CURSO DE GESTÃO DE REDES - UNIFACS – UNIVERSIDADE SALVADOR. Salvador, Bahia, Brasil.

Neto, B. B., Monteiro, P. d., & Queiroga, S. L. (2012). Aplicabilidade dos Microcontroladores em Inovações Tecnológicas. Palmas, Tocantins, Brasil.

Okuda, F. M. (2008). Bluetooth, Trabalho de conclusão de curso da Universidade de São Paulo. São Paulo.

Penido, É. d., & Trindade, R. S. (2013). Microcontroladores - IFMG. Ouro Preto, Minas Gerais, Brasil.

Reis, F. d. (24 de Abril de 2015). Introdução aos Microcontroladores. São Paulo, São Paulo, Brasil.

Ribeiro, T. J., & Oliveira, S. C. (2016). Projeto de um Gateway para Automação Residencial. *Revista de Engenharia e Pesquisa Aplicada*, 43-49.

Sampaio, A. F. (2008). SISTEMA DE AUTOMAÇÃO wireless para controle de dispositivos autônomos. Brasília, DF, Brasil.

Santos, W. S., Pereira, M. F., Pereira, R. B., & Neto, E. J. (2016). Miniestacão agrometeorológica baseada na plataforma ESP8266 para aplicações agrícolas, UFMT. Cuiabá, Mato Grosso, Brasil.

Silva, A. O., & Sales, V. M. (2016). Desenvolvimento de um controlador feedback adaptativo para posicionamento de uma antena parabólica receptora - Trabalho de Conclusão do curso Engenharia de Controle e Automação - IFRJ - Fluminense. Campos dos Goytacazes, Rio de Janeiro, Brasil.

Sousa, D. J. (2000). *Desbravando o PIC: Baseado no microcontrolador PIC 16F84. 5ª ed.* São Paulo: Érica.

Sousa, M. B. (2002). *Wireless: Sistemas de rede sem fio.* Rio de janeiro: Brasport.

APÊNDICE A – UPLOAD DO PROGRAMA PARA O ARDUÍNO PRO-MINI

Neste trabalho, foram testadas duas formas para fazer o upload dos códigos para o Arduíno Pro-Mini: por meio do conversor USB-Serial ou por meio de um Arduíno UNO.

- Programando um Arduíno Pro Mini usando conversor FTDI.

A ligação a ser realizada é apresentada na Figura 36 a seguir. Se o conversor utilizado não possuir o pino DTR, como foi o caso do utilizado neste trabalho, o reset deve ser realizado manualmente. Assim, no momento do *upload*, quando o led azul do conversor piscar, deve ser pressionado o botão de *reset* da própria placa do Arduíno Pro-mini, a partir de então o *upload* será iniciado.

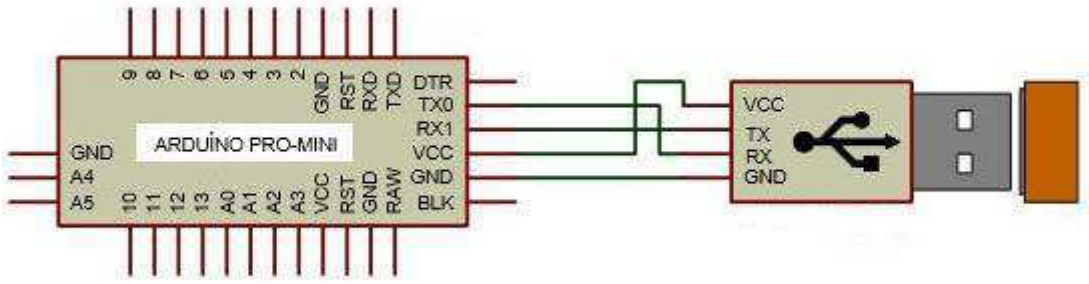


Figura 36 - Ligação do conversor UBS-Serial com o Arduíno Pro-mini.

Realizada a conexão, na IDE, vá no menu FERRAMENTAS, depois em PLACA, e escolha "ARDUÍNO PRO ou PRO MINI (5V, 16 Mhz) W/ ATmega328″. Feito isso, utilize a IDE para carregar normalmente seu programa no Arduíno Mini.

- Programando um Arduíno Pro Mini com Arduíno Uno

Antes de mais nada, é necessário retirar o microcontrolador Atmega do Arduíno Uno, para evitar qualquer tipo de conflito que atrapalhe o funcionamento do circuito.

Retirado o chip, monte o circuito como apresentado na Figura 37. A ligação a ser realizada é apresentada a seguir:

- GRN – Ligado à porta RESET do Arduíno;
- TXD – Ligado ao pino 1 (TX) do Arduíno;
- RX – Ligado ao pino 0 (RX) do Arduíno;
- Vcc – Ligado ao 5 V do Arduíno;
- GND – Ligado ao GND do Arduíno;
- BLK – Não conectado.

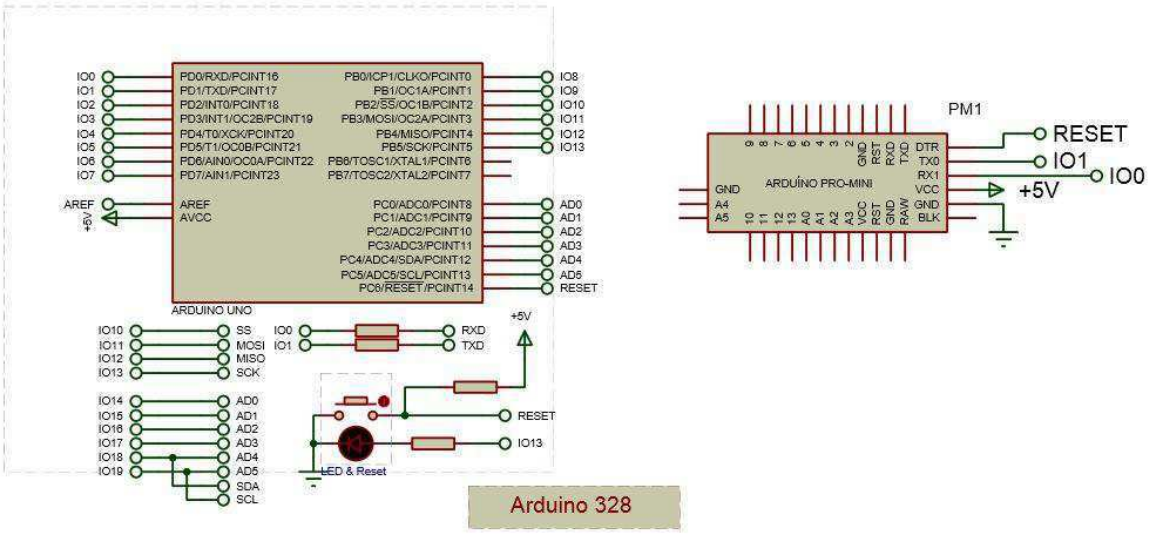


Figura 37 - Conexão do UNO com o Pro-MIni.

Conecte o cabo USB no UNO e abra a IDE do Arduíno. Observe que o Pro-Mini também será ligado, já que está sendo alimentado pelos 5 V do Arduíno Uno. Após escolhida a placa (do mesmo modo que foi descrito anteriormente), utilize a IDE para carregar normalmente seu programa no Arduíno Mini.

APÊNDICE B – PASSOS PARA INSTALAÇÃO DAS PLACAS E BIBLIOTECAS DO ESP8266 PARA A IDE DO ARDUÍNO

Para a utilização da IDE do Arduíno para programar seja o próprio ESP8266 ou outras placas que o tenham como o microcontrolador (como a placa Wemos), é necessário instalar os arquivos da placa ESP8266 no IDE do Arduíno.

Inicialmente certifique-se de que o Arduíno IDE é o mais recente. Em seguida siga os passo a seguir:

Abra as preferências do Arduíno em: Arquivo – Preferências (Figura 38);

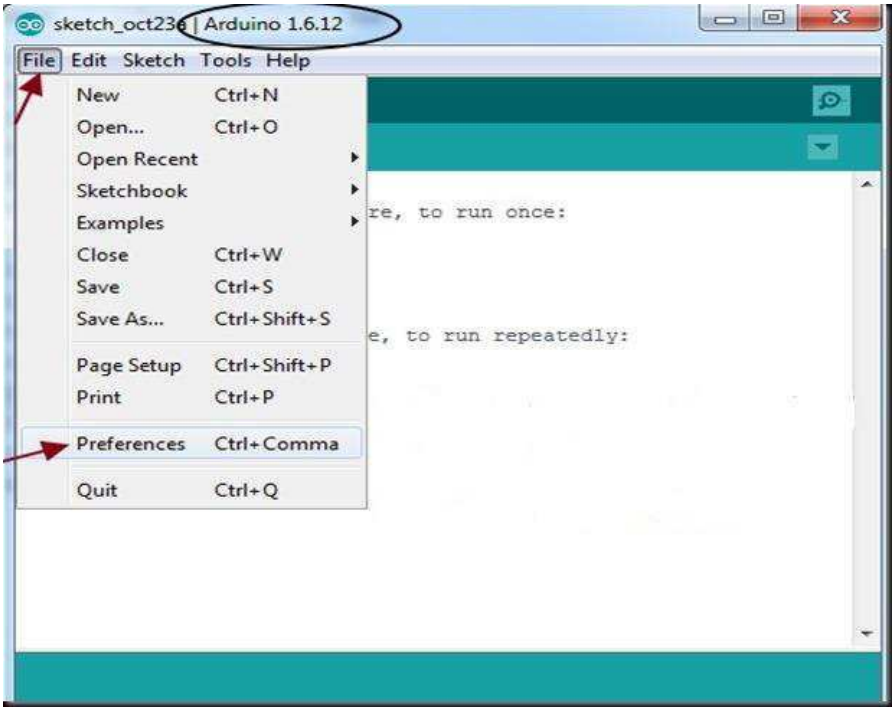


Figura 38 - Aba Aquivo - Preferências na IDE.

- Adicione a seguinte URL aos "*URLs adicionais do Boards Manager*" exibidos na parte inferior da janela Preferências (Figura 39):
  http://arduino.esp8266.com/stable/package_esp8266com_index.json

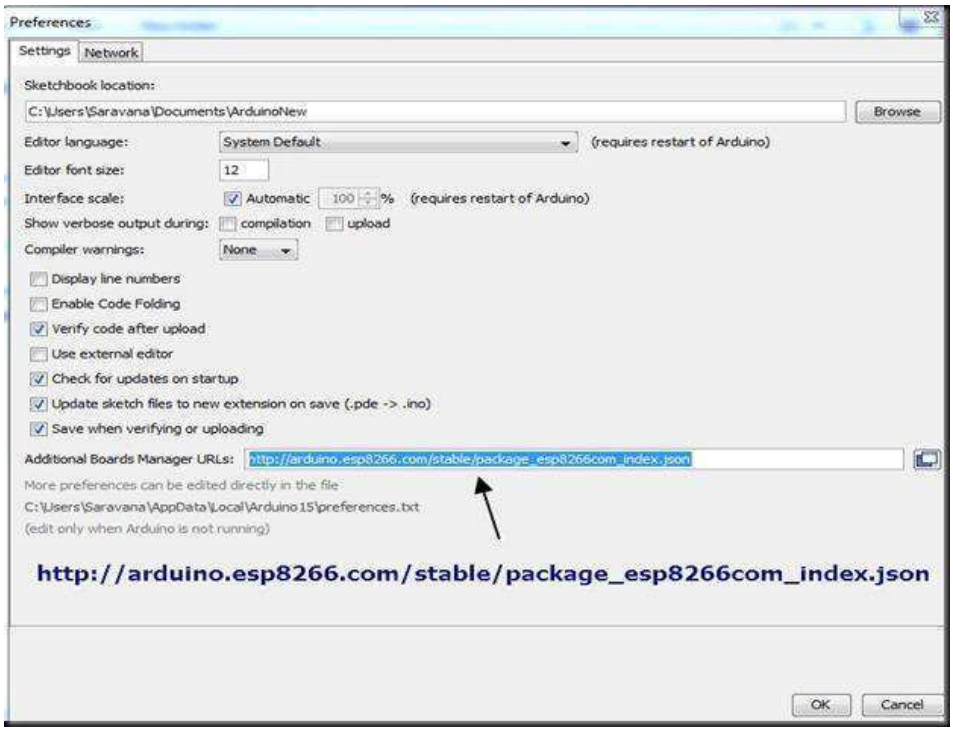


Figura 39 - Inserção da URL para a biblioteca das placas.

- Clique em OK para fechar a janela Preferências. Depois vá em FERRAMENTAS -. BOARDS - BOARDS MANAGER (Figura 40);

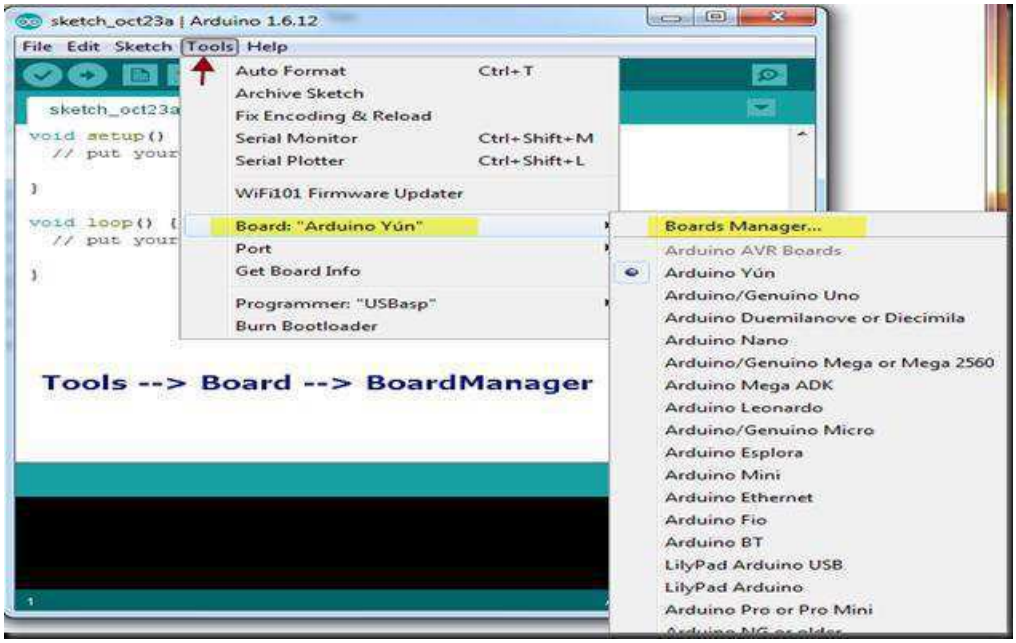


Figura 40 - Aba *Boards Manager* da IDE.

- Quando o *Boards Manager* estiver aberto, vá até a parte inferior. Quando encontrar a placa esp8266, selecione a versão mais recente e clique no botão INSTALL (Figura 41);

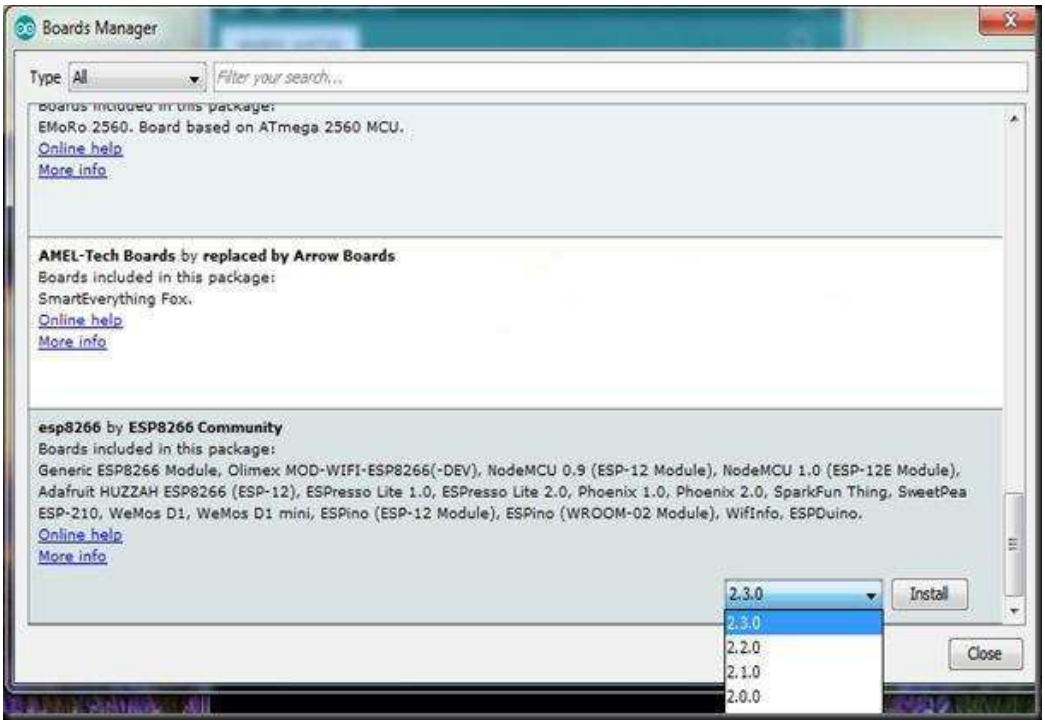


Figura 41 - Instalação da biblioteca das placas ESP.

- Quando a instalação terminar, feche o IDE e inicie novamente. Vá em: Em Ferramentas – Placas, e você poderá ver uma nova lista de Módulos ESP8266 (Figura 42).

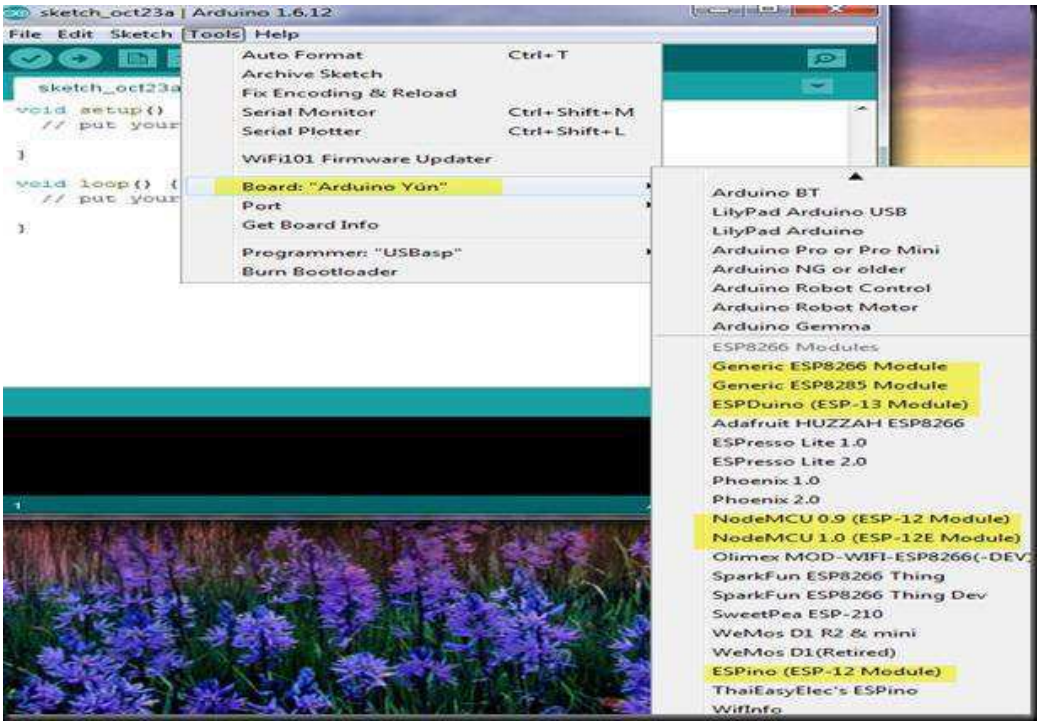


Figura 42 - Placas instaladas na IDE.

APÊNDICE C – CÓDIGO LIGA LED COM O MÓDULO HC-06

```
void setup() {

pinMode(3, OUTPUT);
pinMode(4, OUTPUT);
pinMode(5, OUTPUT);

Serial.begin(9600);
}

void loop() {
  char c = Serial.read();
  if (c == '1')
    digitalWrite(3, HIGH);
  if (c == '2')
    digitalWrite(4, HIGH);
  if (c == '3')
    digitalWrite(5, HIGH);
  if (c == 'A')
    digitalWrite(3, LOW);
  if (c == 'B')
    digitalWrite(4, LOW);
   if (c == 'C')
    digitalWrite(5, LOW);
  delay(1000);
}
```

APÊNDICE D – CÓDIGO LIGA SERVO MOTORES COM O MÓDULO HC-06

```
#include <Servo.h>

Servo myservo1;  // cria um objeto para controle do servo
Servo myservo2;
Servo myservo3;
char rxChar;       //caractere recebido por bluetooth pela serial
int ang = 0;

void setup()
{
   myservo1.attach(2);  // Define que o servo esta ligado a porta 2
  myservo2.attach(3);
  myservo3.attach(4);

  Serial.begin(9600);
}

void loop()
{
  if(Serial.available()){
    rxChar = Serial.read();
  }
  if(rxChar == '1'){
    myservo1.write(ang + 10);
  }
  if(rxChar == 'a'){
    myservo1.write(ang - 10);
  }
   if(rxChar == '2'){
    myservo2.write(ang + 10);
  }
  if(rxChar == 'b'){
    myservo2.write(ang - 10);
  }
   if(rxChar == '3'){
    myservo3.write(ang + 10);
  }
  if(rxChar == 'c'){
    myservo3.write(ang - 10);
  }
}
```

APÊNDICE E – CÓDIGO LIGA SERVO MOTORES E LED'S COM O MÓDULO NRF24L01

```
//Aciona dois servos para cada direção do potenciômetro, liga leds dos dois lados

#include <RF24.h>
#include <Servo.h>

//*************** Controle do RF **********************
#define radioID 1   //Informar "0" para um dispositivo e "1" para o outro dispositivo

struct estruturaDadosRF
{
        boolean ligando = false;    //Esta variavel será usada para solicitar os dados do outro aparelho. Será útil quando o aparelho solicitante esta sendo ligado, para manter os valores do aparelho que já esta ligado.
          boolean botao1 = false;
          boolean botao2 = false;
          int potenciometro1 = 0;
          int potenciometro2 = 0;
};
typedef struct estruturaDadosRF tipoDadosRF;
tipoDadosRF dadosRF;
tipoDadosRF dadosRecebidos;
boolean transmitido = true;
boolean alterado = false;
RF24 radio(7, 8); // Pinos CE e CSN do módulo
byte enderecos[][6] = {"1node", "2node"};

//*************** Controle do Projeto LOCAL ************
boolean botao1Ant = HIGH;
```

```
boolean botao1    = HIGH;
boolean botao2Ant = HIGH;
boolean botao2    = HIGH;
int pot1Ant = 0;
int pot1    = 0;
int pot2Ant = 0;
int pot2    = 0;
int angulo1 = 0;
int angulo2 = 0;
Servo servo1;
Servo servo2;

void setup() {
 //*************** Controle do RF ***********************
 radio.begin();
#if radioID == 0
            radio.openWritingPipe(enderecos[0]);
            radio.openReadingPipe(1, enderecos[1]);
#else
            radio.openWritingPipe(enderecos[1]);
            radio.openReadingPipe(1, enderecos[0]);
#endif

//Solicita os dados do outro aparelho, se ele estiver ligado. Tenta a comunicação por 2 segundos.
dadosRF.ligando = true;
radio.stopListening();
 long tempoInicio = millis();
while ( !radio.write( &dadosRF, sizeof(tipoDadosRF) ) ) {
          if ((millis() - tempoInicio) > 2000) {
                    break;
          }
 };
dadosRF.ligando = false;
```

```
radio.startListening();

 //*************** Controle do Projeto LOCAL ************
pinMode(A1, OUTPUT);
pinMode(A2, OUTPUT);
 pinMode(A4, INPUT);
pinMode(A5, INPUT);
pinMode(2, INPUT_PULLUP);
pinMode(3, INPUT_PULLUP);
servo1.attach(5);
 servo2.attach(6);
}

void loop() {
//*************** Controle do RF ***********************
// se houve alteração dos dados, envia para o outro radio
 if (alterado || !transmitido) {
          radio.stopListening();
           transmitido = radio.write( &dadosRF, sizeof(tipoDadosRF) );
           radio.startListening();
          alterado = false;
}
//verifica se esta recebendo mensagem
if (radio.available()) {
          radio.read( &dadosRecebidos, sizeof(tipoDadosRF) );
          //verifica se houve solicitação de envio dos dados
          if (dadosRecebidos.ligando) {
                    alterado = true;
             } else {
                    dadosRF = dadosRecebidos; }
          }
//*************** Controle do Projeto LOCAL ************
#if radioID == 1
 pot1 = analogRead(A4);
```

```
if (pot1 != pot1Ant) {
        dadosRF.potenciometro1 = pot1;
        alterado = true;  //esta variavel controla o envio dos dados sempre que houver uma alteração}
pot1Ant = pot1;
pot2 = analogRead(A5);
if (pot2 != pot2Ant) {
        dadosRF.potenciometro2 = pot2;
        alterado = true;  //esta variavel controla o envio dos dados sempre que houver uma alteração}
pot2Ant = pot2;
#endif
botao1 = digitalRead(2);
if (botao1 && (botao1 != botao1Ant)) {
        dadosRF.botao1 = !dadosRF.botao1;
        alterado = true;  //esta variavel controla o envio dos dados sempre que houver uma alteração}
botao1Ant = botao1;
botao2 = digitalRead(3);
if (botao2 && (botao2 != botao2Ant)) {
        dadosRF.botao2 = !dadosRF.botao2;
        alterado = true;  //esta variavel controla o envio dos dados sempre que houver uma alteração}
 botao2Ant = botao2;
angulo1 = map(dadosRF.potenciometro1, 0, 1023, 0, 180);
servo1.write(angulo1);
angulo2 = map(dadosRF.potenciometro2, 0, 1023, 0, 180);
servo2.write(angulo2);
digitalWrite(A1, dadosRF.botao1);
digitalWrite(A2, dadosRF.botao2);
delay(10);
}
```

APÊNDICE F – CÓDIGO ACIONA RELÉS COM O MÓDULO NRF24L01

- Transmissor

```
#include <SPI.h>
#include "nRF24L01.h"
#include "RF24.h"

int dados[2]; // Matriz do tipo int chamada dados transmitidos
RF24 radio(7,8);  // Cria um objeto chama radio conectado nos pinos 7 e 8 do Arduino
const uint64_t pipe1 = 0xE8E8F0F0E2LL; //Configurando endereço de comunicação entre os MÓDULOS,

void setup(void)
{
        radio.begin();              // Inicia o modulo para comunicação.
        radio.openWritingPipe(pipe1);
        pinMode(2, INPUT_PULLUP);
        pinMode(5, INPUT_PULLUP);
}
void loop()
{
        dados[0] = digitalRead(5);;
        dados[1] = digitalRead(2);;
        radio.write( dados, sizeof(dados) ); // Envia a Matriz dados
}
```

- Receptor

```
#include <SPI.h>
#include "nRF24L01.h"
#include "RF24.h"

int dados_r[2]; // Matriz do tipo int chamada dados recebidos
RF24 radio(7,8); // Cria um objeto chama radio conectado nos pinos 7 e 8 do Arduino
const uint64_t pipe1 = 0xE8E8F0F0E2LL; //Configurando endereço de comunicação entre os MÓDULOS,

void setup(void)
{
        Serial.begin(9600);           // Inicia a Serial
         radio.begin();               // Inicia o modulo para comunicação
         radio.openReadingPipe(1, pipe1);
        radio.startListening();       // Inicia o modulo para ouvir as requisições.
        pinMode(A0, OUTPUT);
         pinMode(A1, OUTPUT);
}
void loop()
{
        int estado1 = 0;
        int estado2 = 0;
        int flag1;
        int flag1Ant = 0;
        int flag2;
        int flag2Ant = 0;
        if ( radio.available() ) // Verifica se o rádio “ouviu”
        {
                bool done = false;
                 while (!done){
                        done = radio.read( dados_r, sizeof(dados_r) );
                         flag1 = dados_r[0];
```

```
                    if ((flag1 == HIGH) && (flag1Ant == LOW)){
                        estado1 = 1 - estado1;
                        delay (20);
                    }
                    flag1Ant = flag1;
                    if(estado1 == 1)
                        digitalWrite(A0, HIGH);
                    else
                        digitalWrite(A0, LOW);
                    flag2 = dados_r[1];
                    if ((flag2 == HIGH) && (flag2Ant == LOW)){
                        estado2 = 1 - estado2;
                        delay (20);
                    }
                    flag2Ant = flag2;
                    if(estado2 == 1)
                        digitalWrite(A1, HIGH);
                    else
                        digitalWrite(A1, LOW);
                    Serial.print("Valor A0:  ");
                    Serial.print(dados_r[0]);
                    Serial.print("  Valor A1:  ");
                    Serial.print(dados_r[1]);
                    Serial.println("");
                }
            }
            else
            {
               Serial.print("Nenhum dado recebido");
            }
            delay(15); // Delay para atualização
}
```

APÊNDICE G – CÓDIGO LIGA LEDS COM O HC-12

```
//Programa : Modulo Wireless HC-12 e Arduino - envia do PC

char buf; //Armazena o caracter recebido
//Define os pinos dos leds
int pinoled_verm = A0;
int pinoled_amar = A1;
int pinoled_azul = A2;
//Armazena o estado dos pinos
int estadoled_verm = 0;
int estadoled_amar = 0;
int estadoled_azul = 0;

void setup()
{
        //Define os pinos dos leds como saida
         pinMode(pinoled_verm, OUTPUT);
         pinMode(pinoled_amar, OUTPUT);
         pinMode(pinoled_azul, OUTPUT);
         Serial.begin(9600);
}
void loop()
{
         buf = Serial.read();
           if (buf == '1') //Led vermelho
          {
                estadoled_verm = !estadoled_verm; //Inverte o estado do led
        (HIGH/LOW)
                digitalWrite(pinoled_verm, estadoled_verm); //Aciona a porta do led
                mySerial.println("Estado do led vermelho alterado!");      //Responde com
msg sobre o estado do led
```

```
        }
        if (buf == '2') //Led verde
        {
                estadoled_amar = !estadoled_amar;
                digitalWrite(pinoled_amar, estadoled_amar);
                 mySerial.println("Estado do led amarelo alterado!");
        }
        if (buf == '3') //Led azul
        {
                estadoled_azul = !estadoled_azul;
                digitalWrite(pinoled_azul, estadoled_azul);
                mySerial.println("Estado do led azul alterado!");
          }
}
```

APÊNDICE H – CÓDIGO CONTROLE DO BRILHO DO LED POR SINAL PWM COM O HC-12

- Transmissor:

```
int analogPin = 0; // pino para leitura do potenciômetro

void setup() {
        Serial.begin(9600);
}
void loop() {
         Serial.write((analogRead(analogPin))/4);
}
```

- Receptor

```
int ledPin = 3; // pino do led
byte val; //valor recebido pela serial
void setup() {
        pinMode(ledPin, OUTPUT); // configura pino como saída
        Serial.begin(9600);
}
void loop() {
        val = Serial.read();
        Serial.println(val);
         analogWrite(ledPin, val); // aciona led com o valor analógico lido
                                   //dividido por 4 para ajustar ao valor
                                   //máximo que pode ser atribuído a função
}
```

APÊNDICE I – CÓDIGO WEMOS: CONEXÃO COM A REDE WIFI

```
#include <ESP8266WiFi.h> // Importa a Biblioteca ESP8266WiFi
const char* ssid = "ssid"; // SSID da Rede
const char* password = "password"; // Senha da Rede
WiFiServer server(80); // “Porta do roteador”

void setup() {
        Serial.begin(115200);
        delay(10);
        Serial.println();
        Serial.println();
        Serial.print("Conectando-se em: ");
        Serial.println(ssid);
        WiFi.begin(ssid, password); // Conecta na Rede Wireless
        while (WiFi.status() != WL_CONNECTED) {
                delay(500);
                Serial.print("."); }
        Serial.println("");
        Serial.print("Conectado na Rede: ");
        Serial.println(ssid);
        server.begin();//Start the server
        Serial.println("Server started");
        Serial.print("Use essa URL para conectar: "); //Print the IP address
        Serial.print("http://");
        Serial.print(WiFi.localIP()); // Mostra o IP que foi atribuido para o ESP8266
        Serial.println("/");
}
void loop() {
// put your main code here, to run repeatedly }
```

APÊNDICE J – CÓDIGO LIGA LEDS COM A PLACA WEMOS

```
#include <ESP8266WiFi.h>
const char* ssid = "ssid";
const char* password = "password";
int ledPin = D5;
WiFiServer server(8082);

void setup() {
	Serial.begin(115200);
	delay(10);
	 pinMode(ledPin, OUTPUT);
	digitalWrite(ledPin, LOW);
	// Connect to WiFi network
	Serial.println();
	Serial.println();
	Serial.print("Connecting to ");
	Serial.println(ssid);
	WiFi.begin(ssid, password);
	while (WiFi.status() != WL_CONNECTED) {
		delay(500);
		Serial.print(".");
	}
	Serial.println("");
	Serial.println("WiFi connected");
	server.begin();// Start the server
	Serial.println("Server started");
	Serial.print("Use this URL : "); // Print the IP address
	Serial.print("http://");
	Serial.print(WiFi.localIP());
```

```
        Serial.println("/");
}
void loop() {
        // Check if a client has connected
        WiFiClient client = server.available();
        if (!client) {
                return;
        }
        // Wait until the client sends some data
        Serial.println("new client");
        while(!client.available()){
                delay(1);
        }
        // Read the first line of the request
        String request = client.readStringUntil('\r');
        Serial.println(request);
        client.flush();
        // Match the request
        int value = LOW;
        if (request.indexOf("/LED=ON") != -1) {
                digitalWrite(ledPin, HIGH);
                value = HIGH;
        }
        if (request.indexOf("/LED=OFF") != -1){
                digitalWrite(ledPin, LOW);
                value = LOW;
        }
        // Return the response
        client.println("HTTP/1.1 200 OK");
        client.println("Content-Type: text/html");
        client.println(""); //  do not forget this one
        client.println("<!DOCTYPE HTML>");
        client.println("<html>");
```

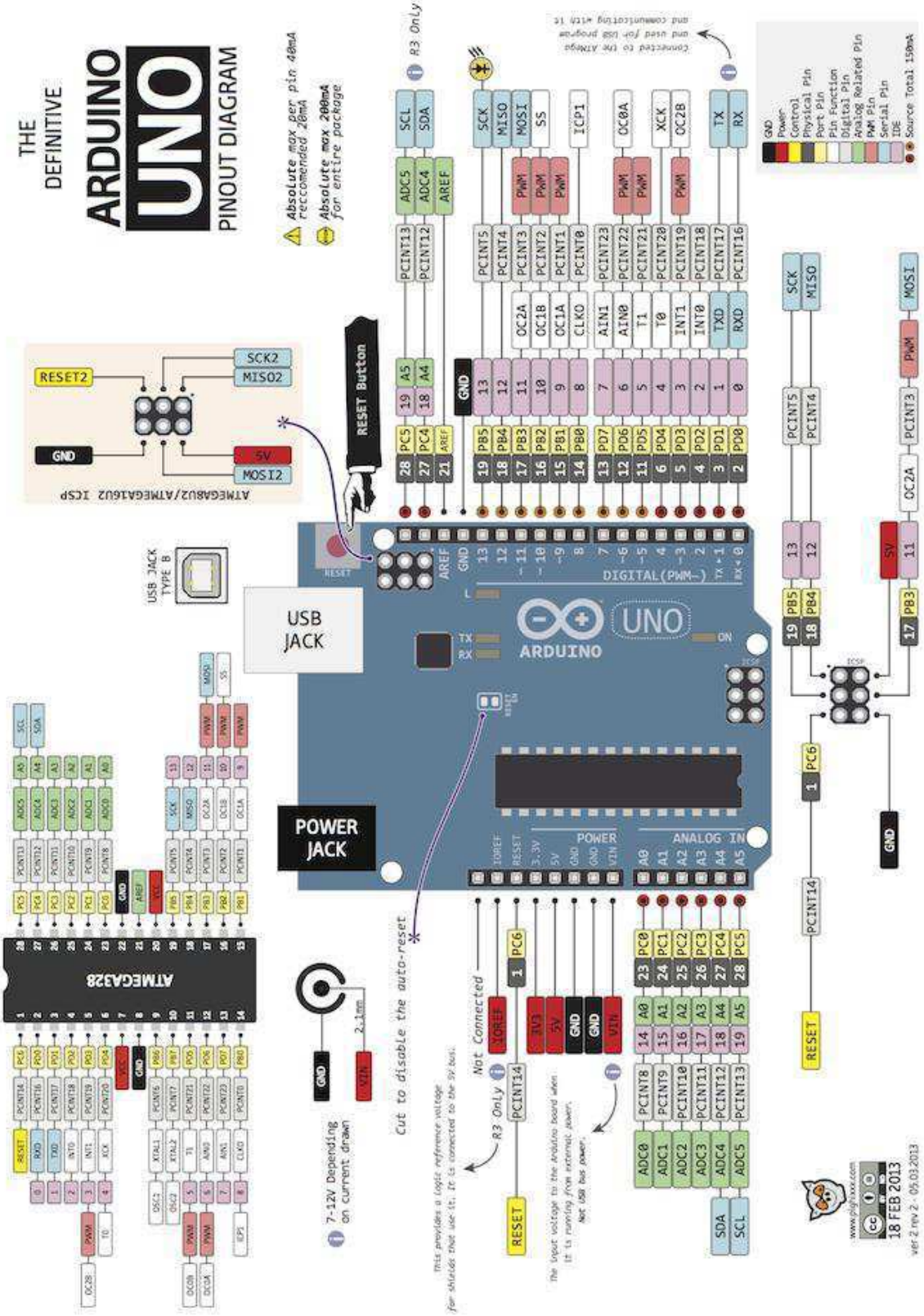
```
        client.print("Led pin is now: ");

        if(value == HIGH) {
                client.print("On");
        } else {
                client.print("Off");
        }
        client.println("<br><br>");
        client.println("Click <a href=\"/LED=ON\">here</a> turn the LED on pin 5 ON<br>");
        client.println("Click <a href=\"/LED=OFF\">here</a> turn the LED on pin 5 OFF<br>");
        client.println("</html>");

        delay(1);
        Serial.println("Client disconnected");
        Serial.println("");

}
```

ANEXO A – MAPA DE PINOS DO ARDUÍNO UNO



ANEXO B – MAPA DE PINOS DO ARDUÍNO PRO-MINI