INSTALAÇÃO DO SISTEMA DE TOQUE IFRN

1. Introdução

Diante da inspiração de criar uma sineta automática no campus avançado Parelhas, com o intuito de melhorar a pontualidade dos horários de aulas, turnos e a assiduidade dos alunos, professores e servidores, foi implantado um sistema com Arduino que pudesse atender a esta demanda. Este manual tem como objetivo documentar e tornar possível a reprodução deste projeto.

Arduino é uma placa micro-controladora programável e de fácil utilização, podendo ser ajustada de acordo com a necessidade do usuário. Pode ser conectada a um computador e programada via IDE utilizando uma linguagem baseada em C/C++ (Wiring).

Neste projeto foi utilizado 1(um) Arduino Uno como placa controladora, 1 (um) shield de ethernet (modelo hanrun HR911105A) para a obtenção de hora atual pela rede via NTP (protocolo para sincronização dos relógios), 1 (um) relé para chavear o circuito acionando a sineta (modelo SRD-05VDC-SL-C), relógio DS3231 (modelo para raspberry), 7 (sete) cabos jumpers para interligar componentes, 1(um) cabo USB tipo B para USB tipo A para carregar o código fonte do sistema. Pode-se ver de forma mais detalhada a lista de componentes abaixo:

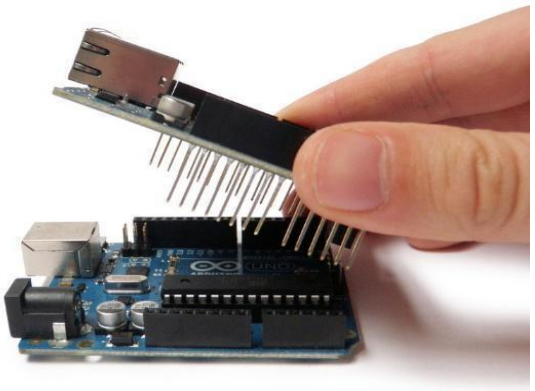
Componentes

- Shield de ethernet para arduino hanrun HR911105A;
- Relé SRD-05VDC-SL-C;
- Clock DS3132 (modelo para raspberry);
- Arduino Uno ;
- Jumpers macho fêmea para arduino;
- Cabo USB 2.0 A para USB 2.0 B;
- Fonte para arduino 9 volts;
- Fios de cores variadas de bitola 1,5mm² (tamanho vai depender da distância entre o sistema e a sineta).

2. Como montar

Acople o shield Ethernet no Arduino Uno. Conforme pode ser visto na Figura 1:

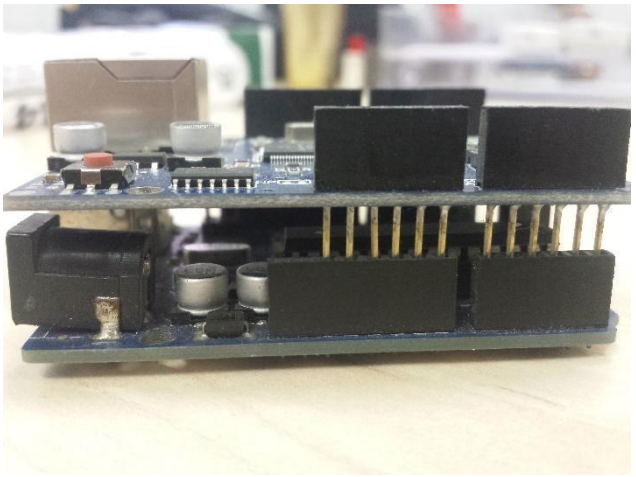
Figura 1: Acoplando SHIELD Ethernet HR911105A no Arduino Uno.



Fonte: Seria Link

De forma de que ambos fiquem com a seguinte disposição:

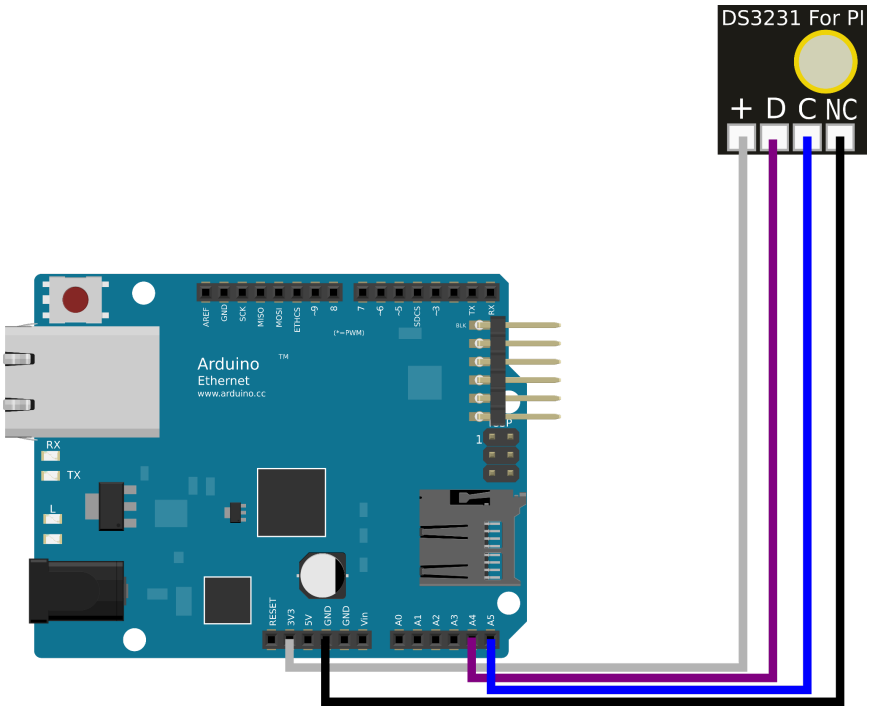
Figura 2: Arduino Uno com SHIELD Ethernet HR911105A acoplado.



Fonte: autoria própria.

Em seguida, deve ser instalado o relógio DS3231, cuja conectividade de fios deve ser feita da seguinte forma:

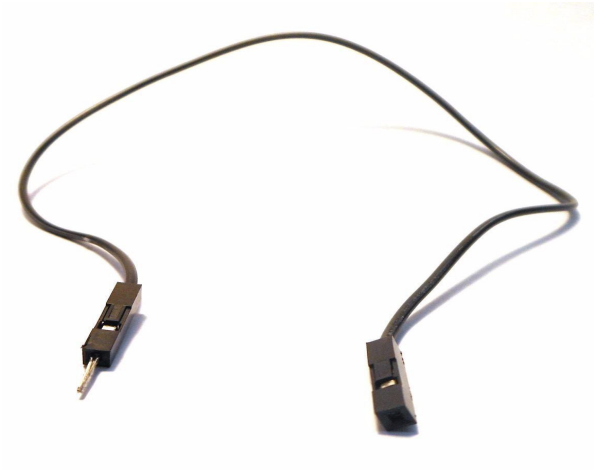
Figura 3: Esquema Shield Ethernet com o DS3231



Fonte: autoria própria.

Esta conexão é feita através de jumpers, que são fios como os da Figura 4. Esta ligação possibilitará o fornecimento de energia ao relógio e troca de dados entre ele e o Arduino.

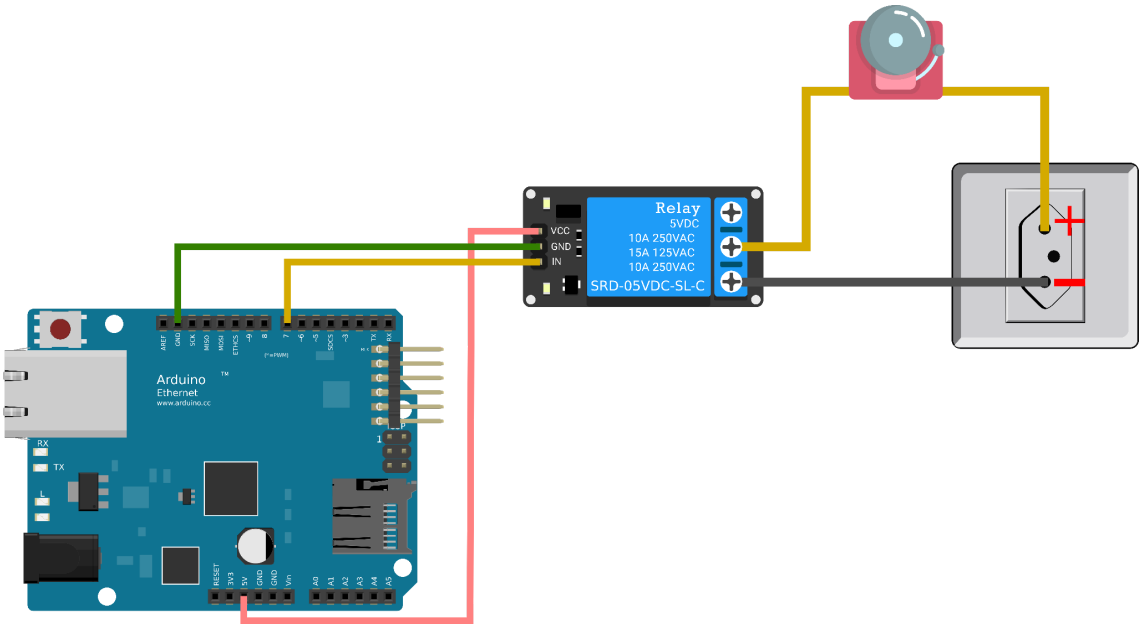
Figura 4: Jumper macho fêmea.



Fonte: albatronic componentes eletrônicos.

O próximo passo de instalação é o relay de 5 volts. Um relay é como um interruptor mecânico (utilizado em residências para o acendimento de lâmpadas). A diferença é que o relay é acionado pelo arduino, utilizando de um sinal elétrico. Neste projeto, este aparelho deve ser conectado ao arduino conforme a Figura 5.

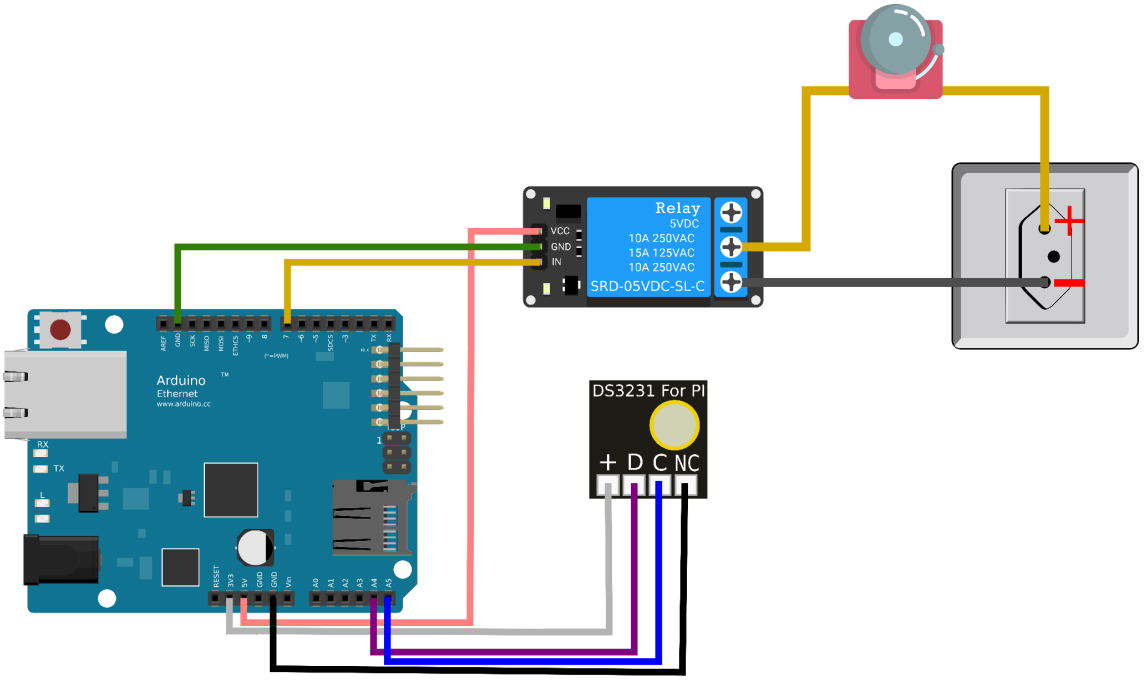
Figura 5: Esquema Arduino Uno com o relé (SRD-05VDC-SL-C) ligado no com a sineta desenvolvido com fritzing.



Fonte: autoria própria.

Os fios amarelo e preto deveriam ser de bitola 1,5mm, devido à passagem de tensões mais altas. Os fios do Arduino devem, por sua vez, ficar conectados no 5V, GND e 7. Pode-se verificar quais as equivalências de conexão no SRD-05VDC-SL-C através da Figura 6. Um detalhe importante de se destacar é que este tipo de relay opera em alta, o que significa que a passagem de energia nos fios de alta tensão é interrompida quando o pino IN é acionado. Desta forma, se houver mudança na conectividade desta entrada, deve-se utilizar portas de sinal constante do Arduino, para evitar que o relay feche circuito devido a pequenas variações na tensão, fazendo com que a sineta possa ficar tocando de forma intermitente.

Figura 6: Esquema de todos os componentes dispostos em conjunto.



Fonte: autoria própria..

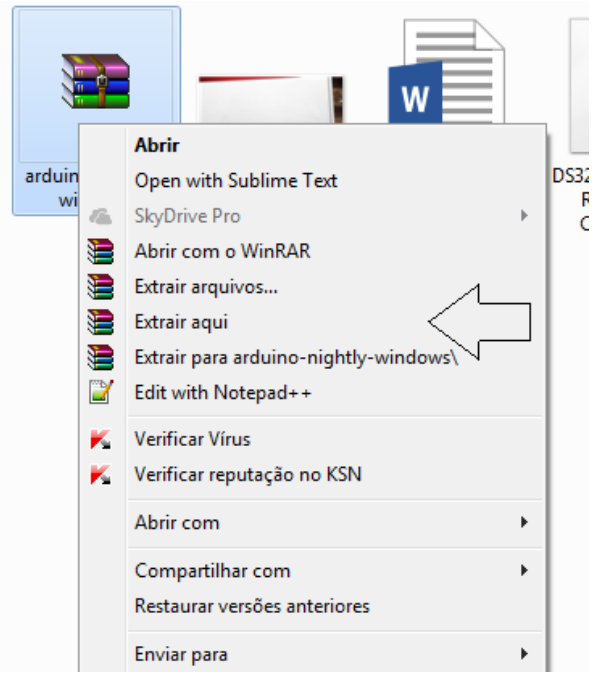
3. Compilando o código e aplicando no Arduino

O código-fonte está disponível no github (https://github.com/marcocspc/ToqueIFRNPAAS), o qual deve ser carregado no Arduino. O componente principal encontra-se na pasta Cliente/src/main.h. O usuário é convidado a ler o algoritmo caso deseje fazer alterações de horário ou mudanças na rotina desenvolvida.

Para que a aplicação do código na placa seja feita, a IDE do Arduino deve ser baixada no seguinte site: https://www.arduino.cc/en/Main/Software.

Após o download, o arquivo .zip pode ser descompactado para que se possa ter acesso ao aplicativo.

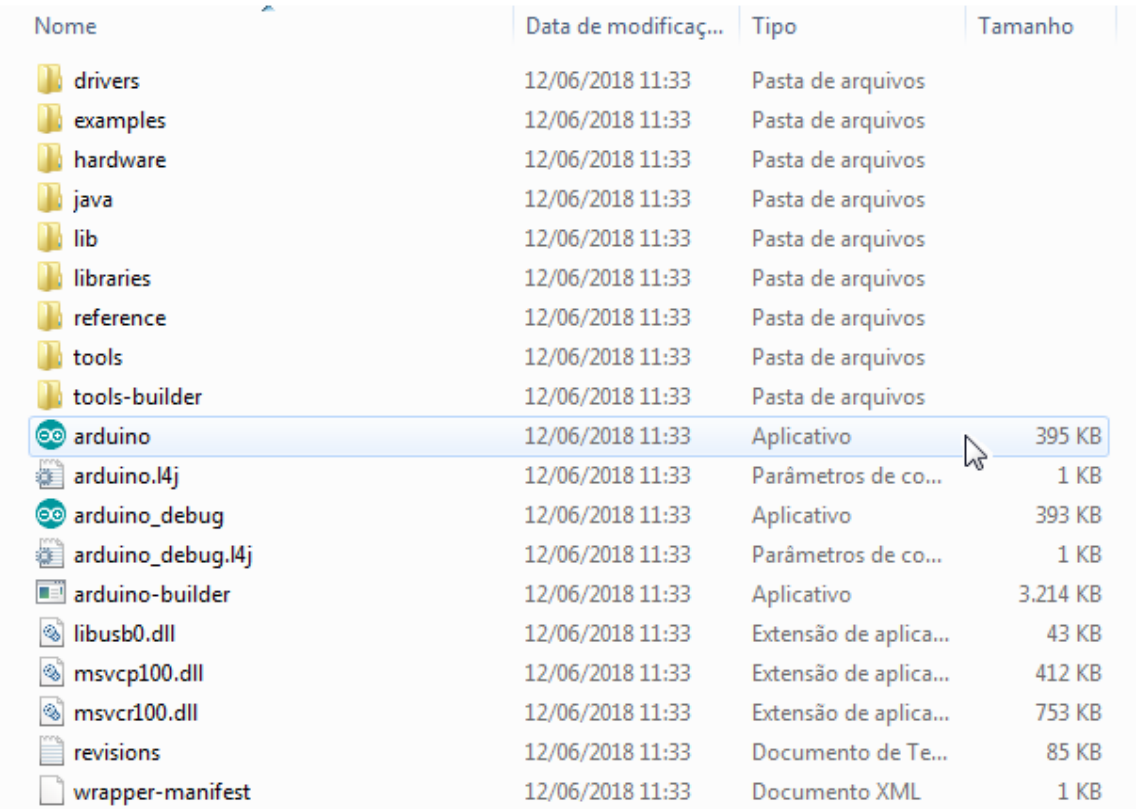
Figura 7: Opções após clicar no botão direito do mouse sobre o arquivo .zip.



Fonte: autoria própria..

Depois de descompactado, o programa pode ser encontrado na pasta criada, conforme demonstrado na Figura 8.

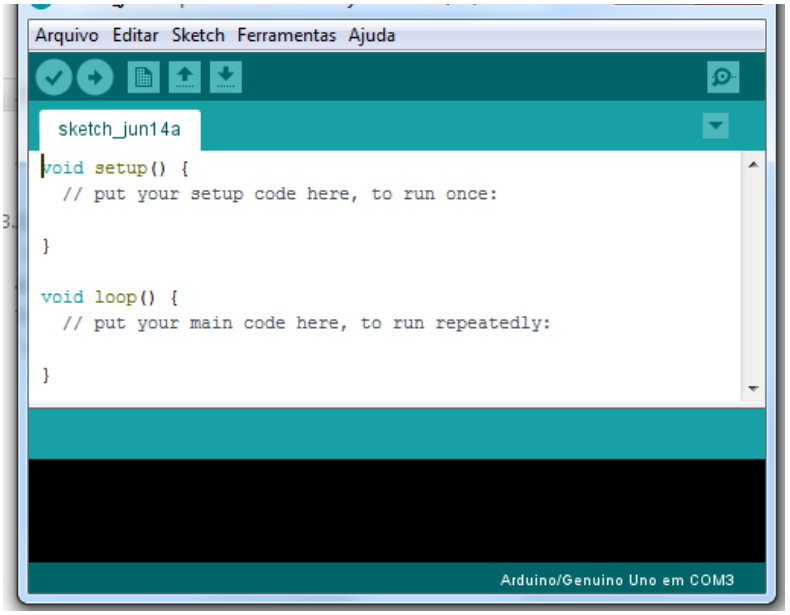
Figura 8: Aplicativo Arduino em destaque, executável.



Fonte: autoria própria..

O programa, após carregado, cria uma janela como a da Figura 9.

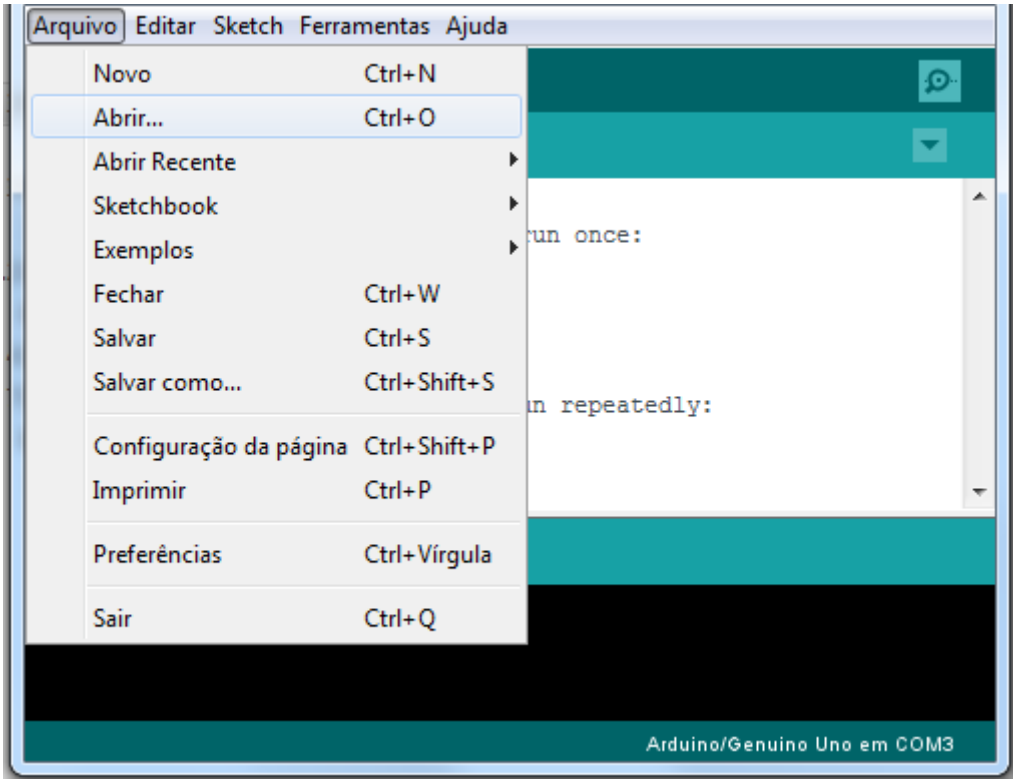
Figura 9: IDE para programar Arduino.



Fonte: autoria própria.

Deve-se, inicialmente, abrir o código. Para isto pode-se clicar em Arquivo -> Abrir e seguir para o diretório onde o código fonte foi baixado, conforme demonstrado na Figura 10.

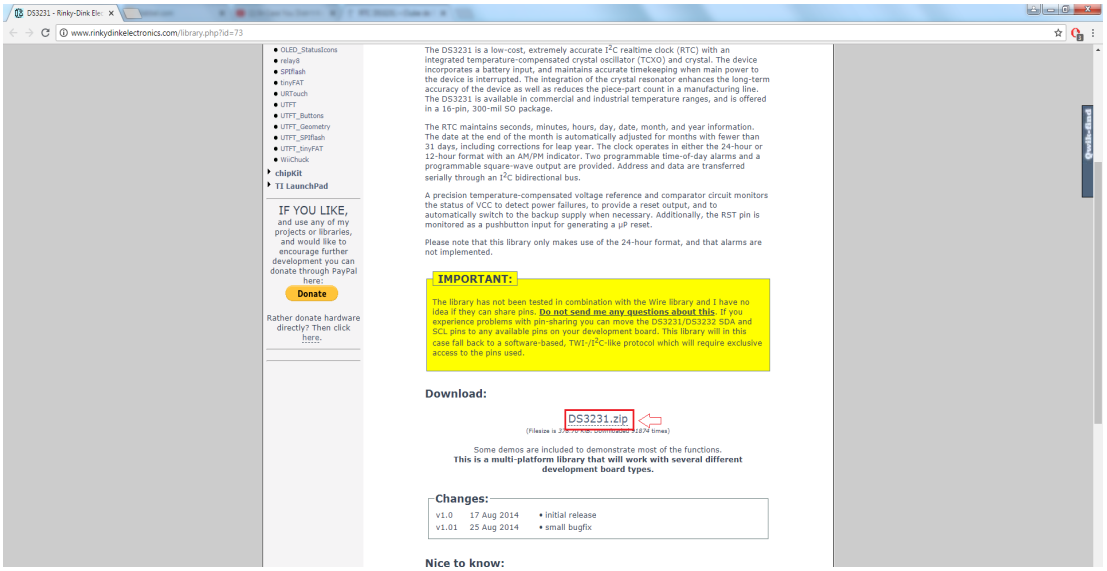
Figura 10: IDE para programar Arduino.



Fonte: autoria própria.

Após isso, é necessário o download da biblioteca capaz de comunicar-se com o relógio DS3231. Tal operação pode ser feita através do acesso ao seguinte site: http://www.rinkydinkelectronics.comlibrary.phpid=73.

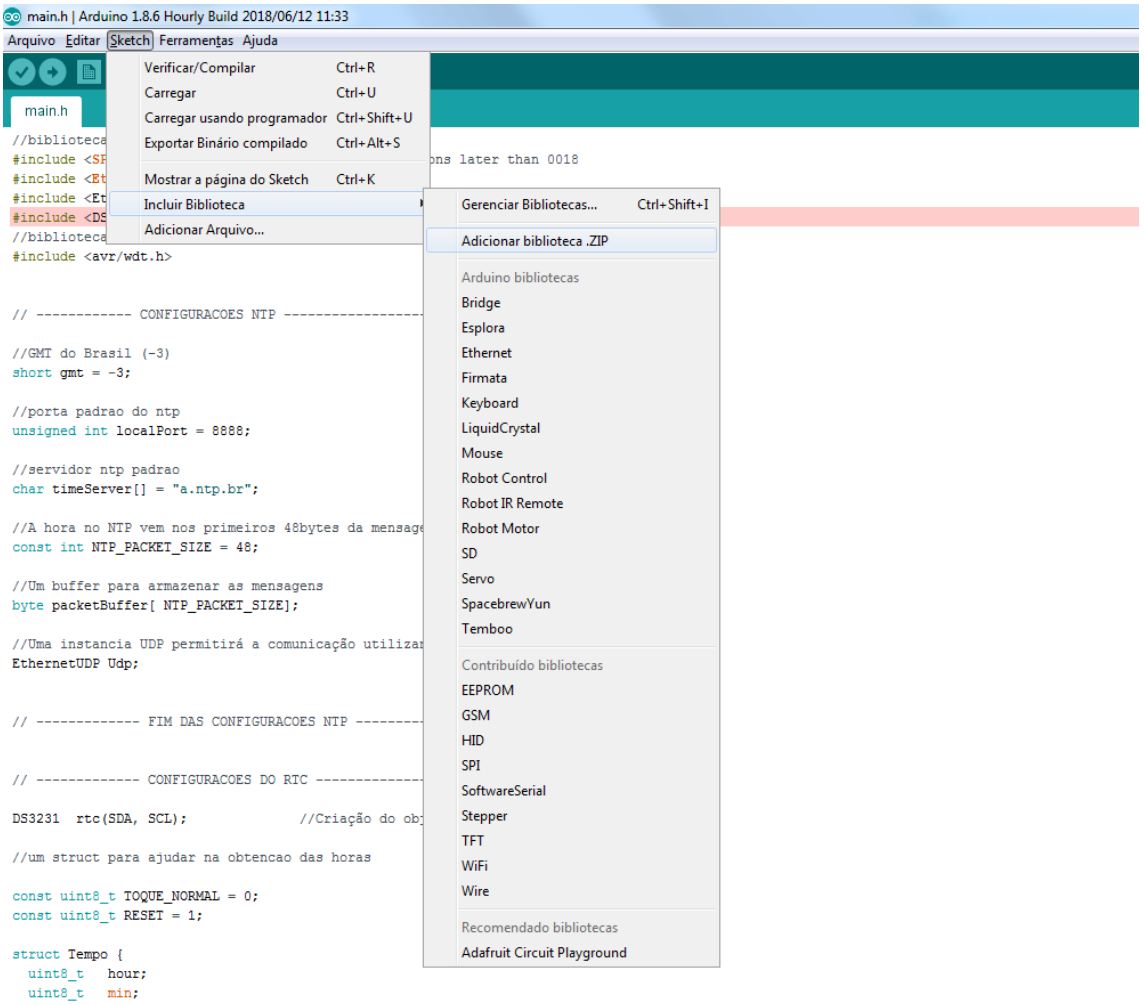
Figura 11: Site da biblioteca do shield DS3231.



Fonte: autoria própria.

Diferentemente da IDE do Arduino, a biblioteca não deve ser extraída. Ela deve ser incluída no aplicativo que irá carregar o programa para o Arduino. Isto pode ser feito clicando em “Adicionar biblioteca .ZIP”, conforme mostra a Figura 12. Em seguida, aparecerá o explorador de arquivos do Windows, o usuário pode localizar a biblioteca baixada e adicioná-la.

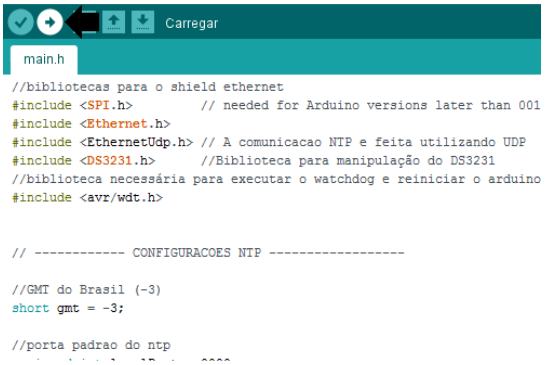
Figura 12: IDE para programar Arduino.



Fonte: autoria própria.

Por fim, a IDE pode ser usada para carregá-lo. Para isso, o arduino deverá ser conectado ao computador com um cabo USB A para USB B. Em seguida, clicando na seta localizada no canto superior esquerdo da IDE, o código é compilado e enviado à placa programável. Como na Figura 13:

Figura 13: IDE para programar arduino.



Fonte: autoria própria.

Após isso, o circuito pode ser montado no local de destino. Recomenda-se usar uma caixa de sobrepor para organizar os componentes, de forma que a instalação fique menos visível e melhor acabada. Segue o exemplo abaixo na Figura 14:

Figura 14: Caixa de sobrepor.

Fonte: autoria própria..

4. Alterando os horários de toque

Na versão atual, o Cliente Arduino comunica-se pela rede apenas para obter a hora de um servidor através do protocolo NTP. Os horários são informados manualmente no código fonte. Para alteração desses horários é necessária a modificação de duas variáveis: “tamH” e “horarios”.

A variável “tamH” armazena o tamanho do vetor “horarios”, que armazena os dados específicos de cada toque. Portanto, ao se alterar o número de vezes que o alarme irá tocar ao longo do dia “tamH” deve possuir seu valor alterado. Por exemplo, caso sejam 18 toques ao longo do dia “tamH” deverá ter o valor 18. Sua atribuição pode ser encontrada na linha 56 de main.h.ino. Conforme Figura 14:

Figura 14: Atribuição de “tamH”

```
55  //Numero de horarios
56  const unsigned short int tamH = 19;
```

Fonte: autoria própria..

Já “horarios”, por outro lado, trata-se de um vetor de horários. Para alterar seus valores, basta seguir o padrão disponível no próprio código-fonte, conforme a Figura 15.

Figura 15: Atribuição dos valores em “horarios”

```
249
250     //FORMATO {HORA, MINUTO, TEMPO_DE_TOQUE, TEMPO_DE_PAUSA,
251     //         NUMERO_DE_REPETICOES, TIPO}
252
253     //07:00
254     horaAux = {7 , 0, 10, 2, 1, TOQUE_NORMAL};
255     horarios[0] = horaAux;
256
257     //07:45
258     horaAux = {7 , 45, 10, 2, 1, TOQUE_NORMAL};
259     horarios[1] = horaAux;
260
261     //08:30
262     horaAux = {8, 30, 20, 2, 1, TOQUE_NORMAL};
263     horarios[2] = horaAux;
264
265     //08:50
266     horaAux = {8, 50, 20, 2, 1, TOQUE_NORMAL};
267     horarios[3] = horaAux;
268
269     //09:35
270     horaAux = {9, 35, 10, 2, 1, TOQUE_NORMAL};
271     horarios[4] = horaAux;
272
```

Fonte: autoria própria.

Pode ser notado que cada posição do vetor recebe valores enfileirados entre chaves “{}”. Cada valor indica uma propriedade do toque, que segue a seguinte fórmula: {H, mm, st, sp, Nr, T}, onde:

- Ht: Hora do toque, horário em que o alarme deverá ser acionado;
- mm: componente dos minutos na hora. Se a hora for 09:30, o valor dessa variável deverá ser 30.
- st: quantidade de tempo em segundos que o toque permanecerá acontecendo. Por exemplo, para que o toque seja acionado durante 10 segundos, este deverá ser o valor dessa variável.
- sp: quantidade de tempo em segundos que o toque deverá aguardar até recomeçar, caso haja repetições.
- Nr: número de repetições que o toque deverá ter. Em alguns casos é desejado que o toque se repita várias vezes, o número de vezes é informado aqui. O número de segundos em cada pausa é informado em sp.
- T: por fim, o tipo de toque. Estão planejados tipos de incêndio, acidente, etc. No entanto apenas “TOQUE_NORMAL” está implementado e esta deve ser a opção utilizada aqui.