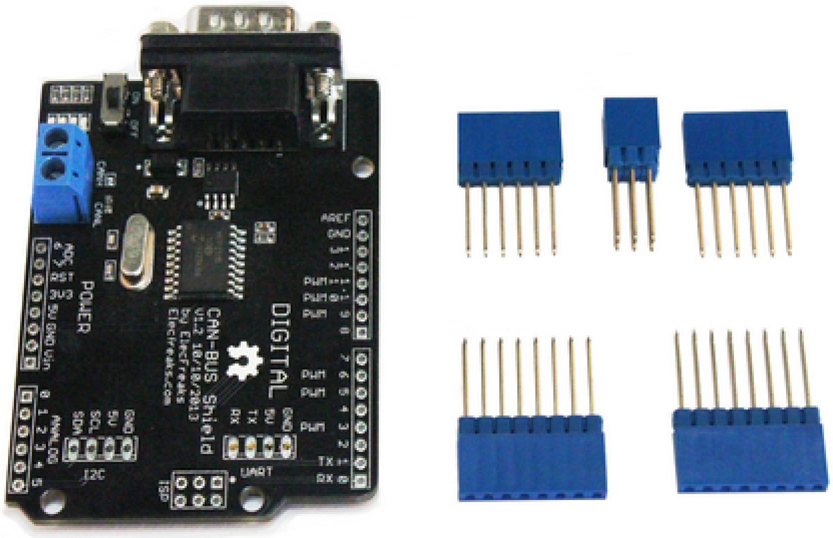
CAN-BUS Shield

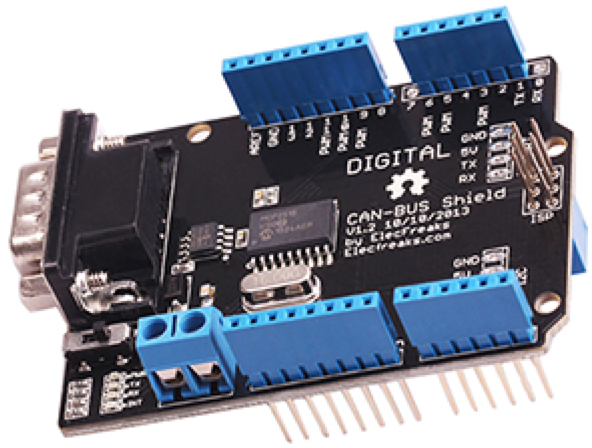
Contents

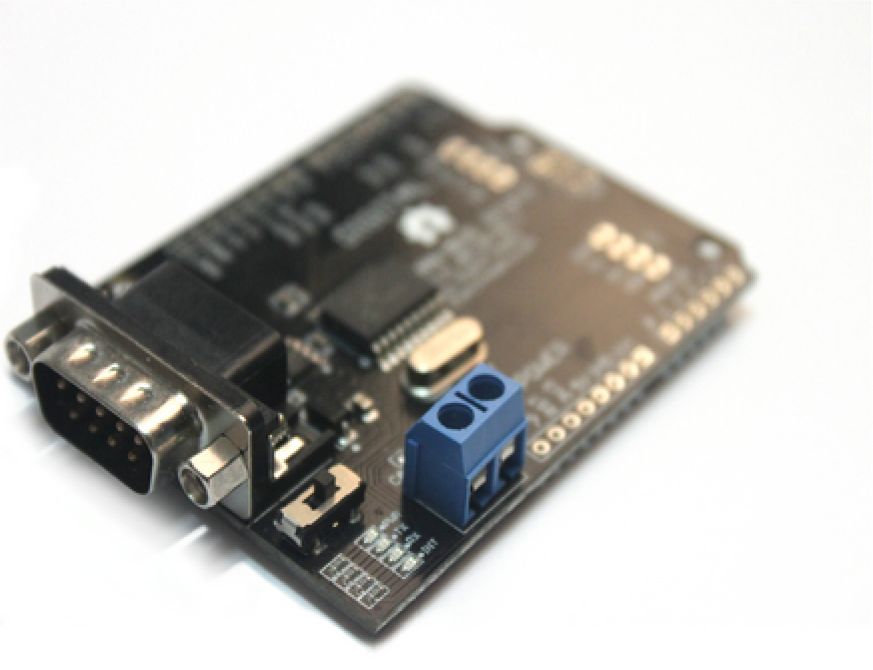
Introduction

The**CAN-BUS** is a common industrial bus because of its long travel distance, medium communication speed and high reliability. It is commonly found on modern machine tools and as an automotive diagnostic bus. This CAN-BUS Shield adopts MCP2515 CAN Bus controller with SPI interface and MCP2551 CAN transceiver to give your Arduino/Seeeduino CAN-BUS capibility. With an OBD-II converter cable added on and the OBD-II library imported, you are ready to build an onboard diagnostic device or data logger.

AUS ELECTRONICS DIRECT COM.AU


DIGITAL
CAN-BUS Shield
V1.2 10/10/2013
by Elecfreaks
Elecfreaks.com
POWER
ANALOG
I2C
UART
ISP
AREF
GND
PWM
TX
RX
RST
3V3
5V
GND
Vin
SDA
SCL


DIGITAL
CAN-BUS Shield
V1.2 10/10/2013
by Elecfreaks
Elecfreaks.com
ISP



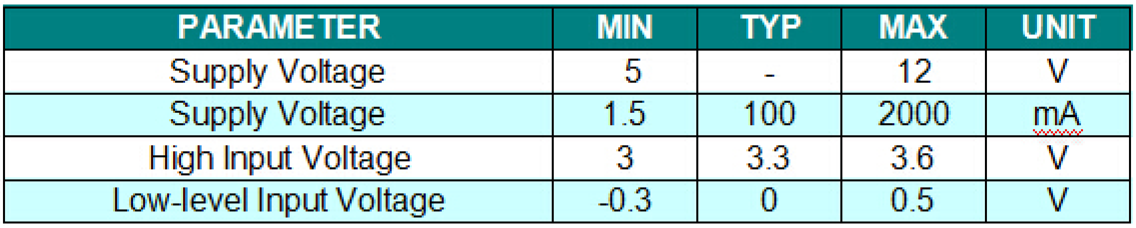
Feature

- Implements CAN V2.0B at up to 1 Mb/s
- SPI Interface up to 10 MHz
- Ariduino/ Freaduino Completely compatible
- Standard (11 bit) and extended (29 bit) data and remote frames
- Industrial standard 9 pin sub-D connector
- Two receive buffers with prioritized message storage
- Operating voltage: DC5-12V
- Size: 78mmx53.5m

Application

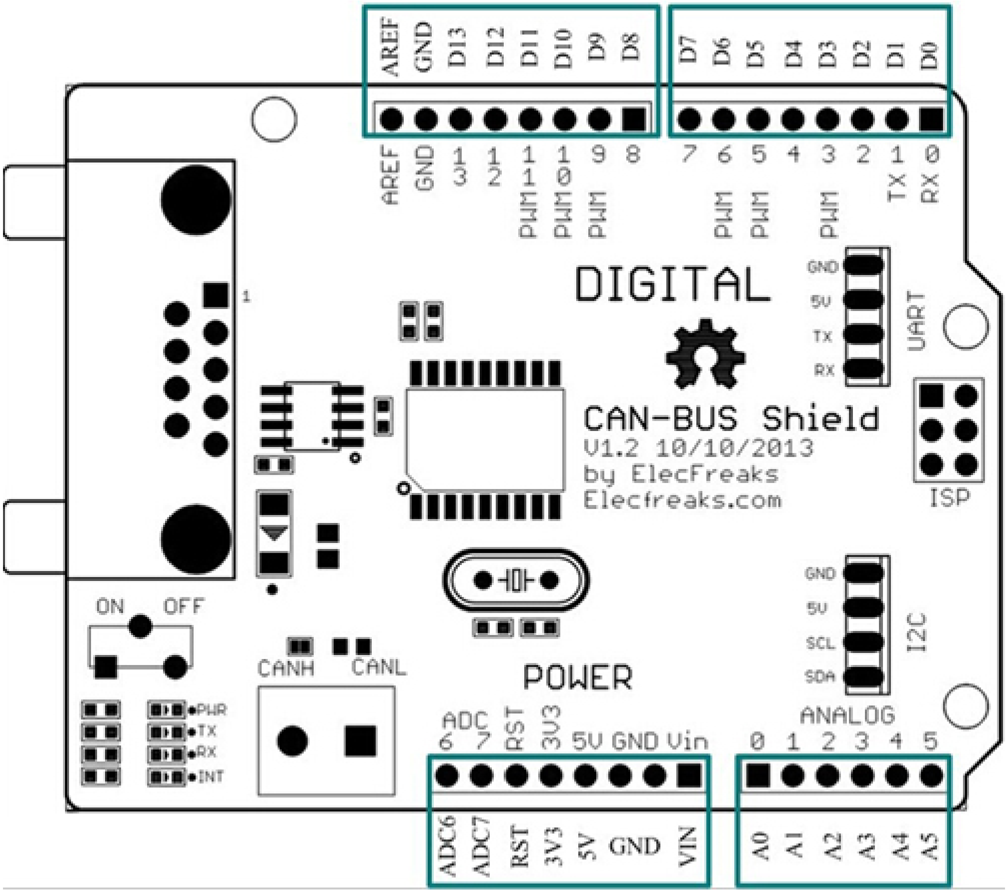
- Industrial control
- Car control field

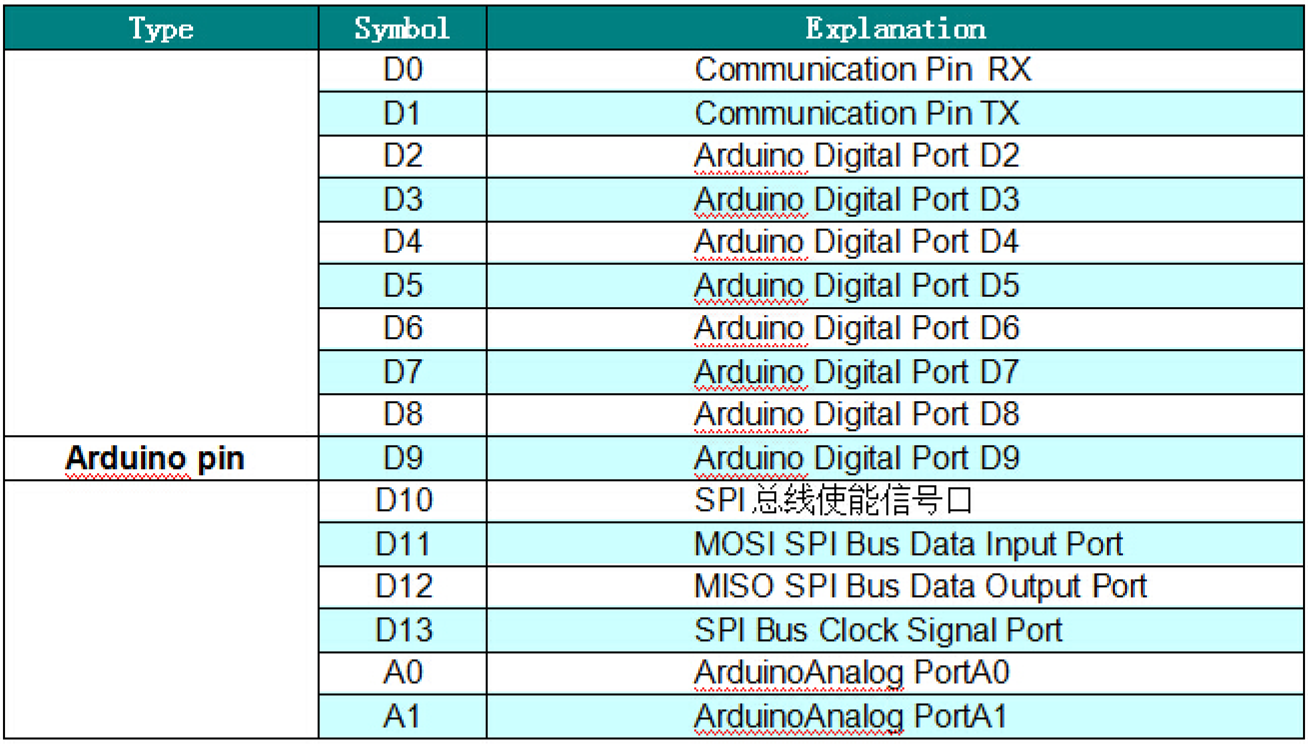
Electronic Properties



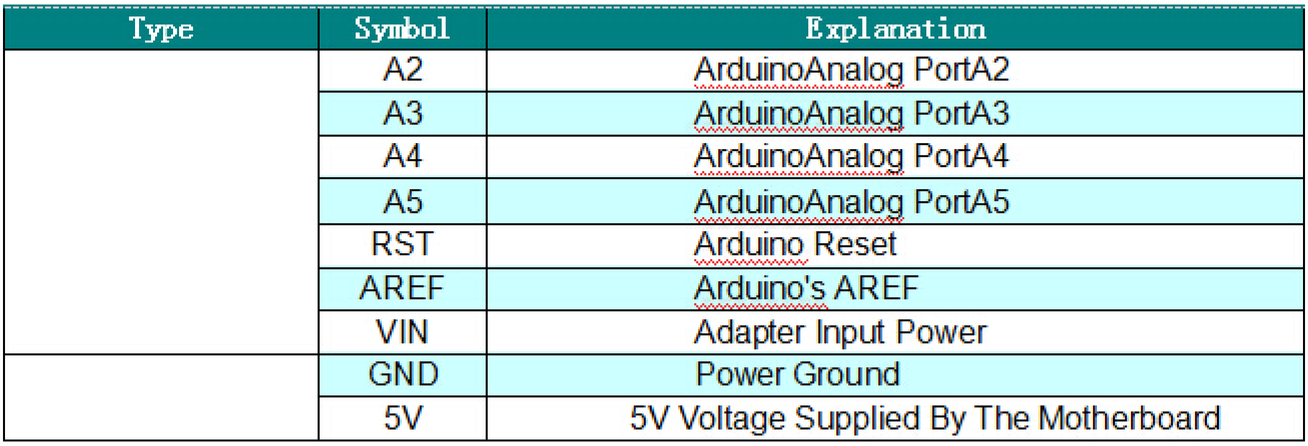
PARAMETER	MIN	TYP	MAX	UNIT
Supply Voltage	5	-	12	V
Supply Voltage	1.5	100	2000	mA
High Input Voltage	3	3.3	3.6	V
Low-level Input Voltage	-0.3	0	0.5	V

Interface Description


AREF
GND
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
AREF
GND
13
12
11 PWM
10 PWM
9 PWM
8
7
6 PWM
5 PWM
4
3 PWM
2
1 TX
0 RX
DIGITAL
GND
5V
TX
RX
UART
CAN-BUS Shield
V1.2 10/10/2013
by ElecFreaks
Elecfreaks.com
ISP
1
ON
OFF
CANH
CANL
PWR
TX
RX
INT
GND
5V
SCL
SDA
I2C
POWER
ANALOG
ADC 6
7
RST
3V3
5V
GND
Vin
0
1
2
3
4
5
ADC6
ADC7
RST
3V3
5V
GND
VIN
A0
A1
A2
A3
A4
A5



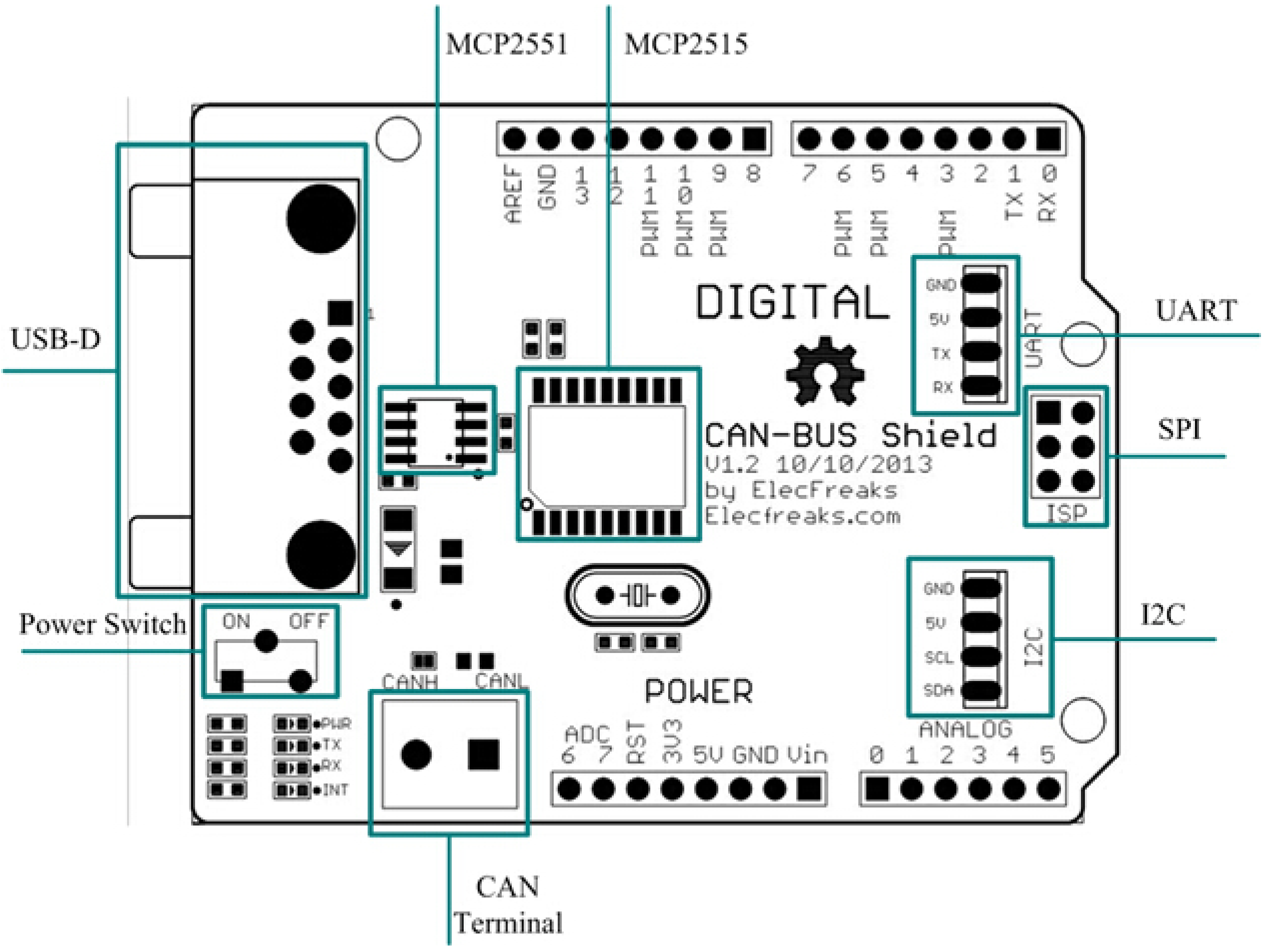
Type	Symbol	Explanation
	D0	Communication Pin RX
	D1	Communication Pin TX
	D2	Arduino Digital Port D2
	D3	Arduino Digital Port D3
	D4	Arduino Digital Port D4
	D5	Arduino Digital Port D5
	D6	Arduino Digital Port D6
	D7	Arduino Digital Port D7
	D8	Arduino Digital Port D8
Arduino pin	D9	Arduino Digital Port D9
	D10	SPI 总线使能信号口
	D11	MOSI SPI Bus Data Input Port
	D12	MISO SPI Bus Data Output Port
	D13	SPI Bus Clock Signal Port
	A0	ArduinoAnalog PortA0
	A1	ArduinoAnalog PortA1



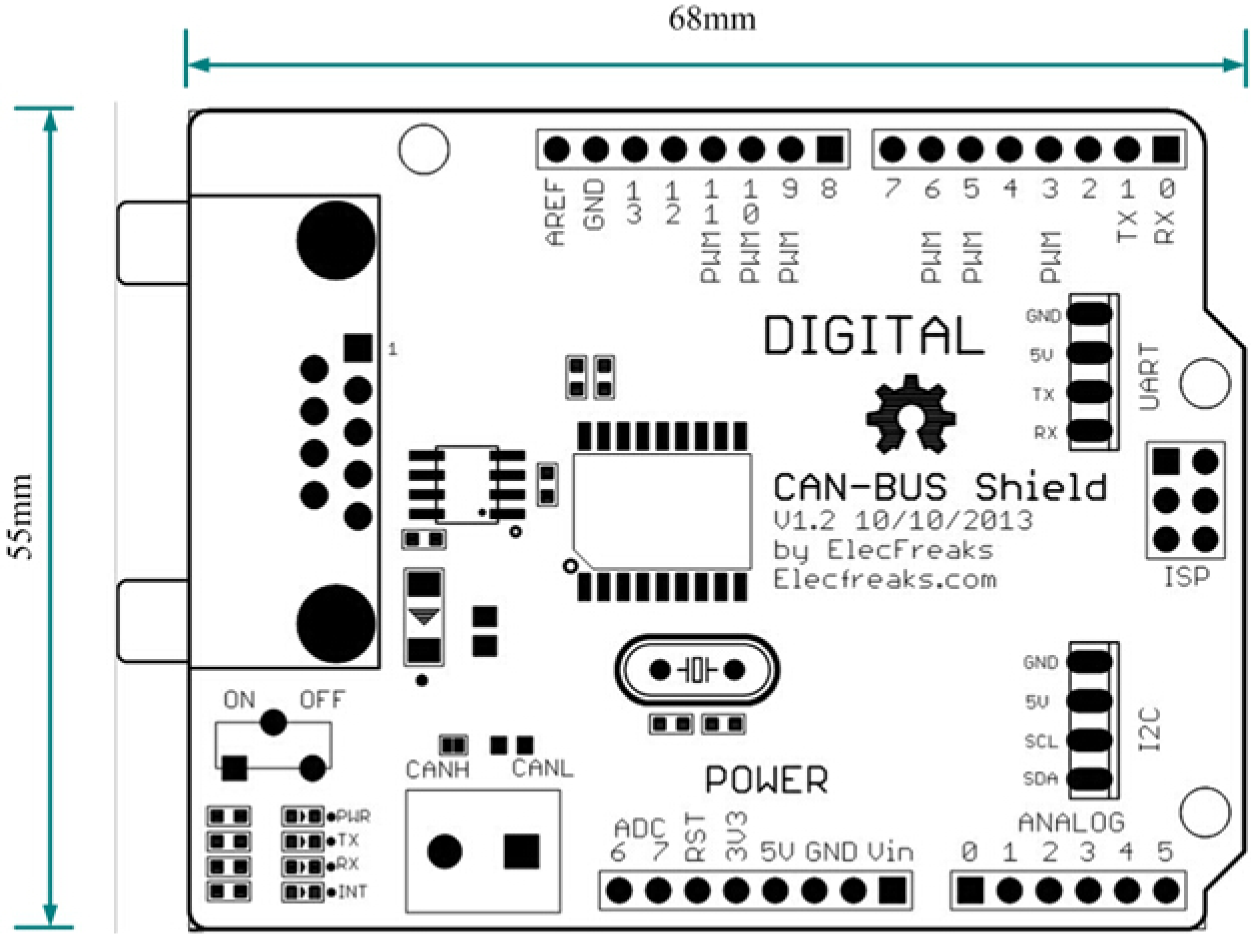
Type	Symbol	Explanation
	A2	ArduinoAnalog PortA2
	A3	ArduinoAnalog PortA3
	A4	ArduinoAnalog PortA4
	A5	ArduinoAnalog PortA5
	RST	Arduino Reset
	AREF	Arduino's AREF
	VIN	Adapter Input Power
	GND	Power Ground
	5V	5V Voltage Supplied By The Motherboard

Specification

Module Description


MCP2551
MCP2515
USB-D
UART
SPI
I2C
Power Switch
CAN Terminal
DIGITAL
CAN-BUS Shield
V1.2 10/10/2013
by ElecFreaks
Elecfreaks.com
POWER
ANALOG
ISP
CANH
CANL
ON
OFF

Dimensions


68mm
55mm

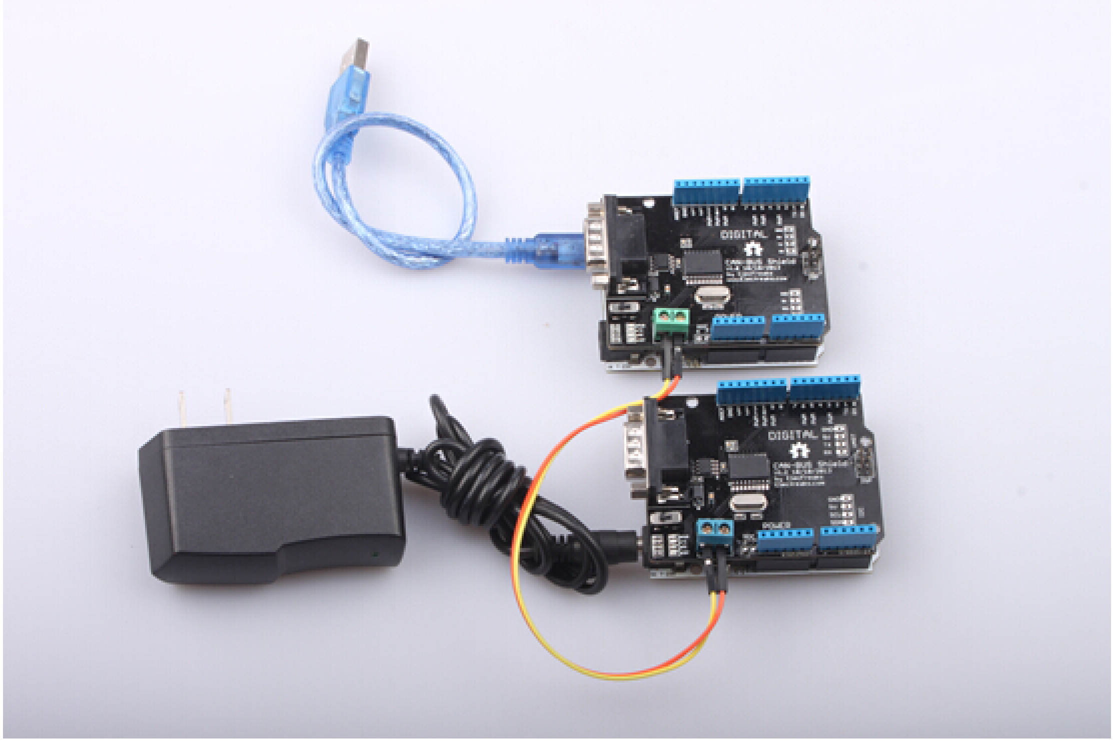
Step 1

Get Tools Prepared:

- UNO *2
- CAN-BUS_Shield *2
- USB *1
- Adapter *1
- Jumper Wire *2

Step 2

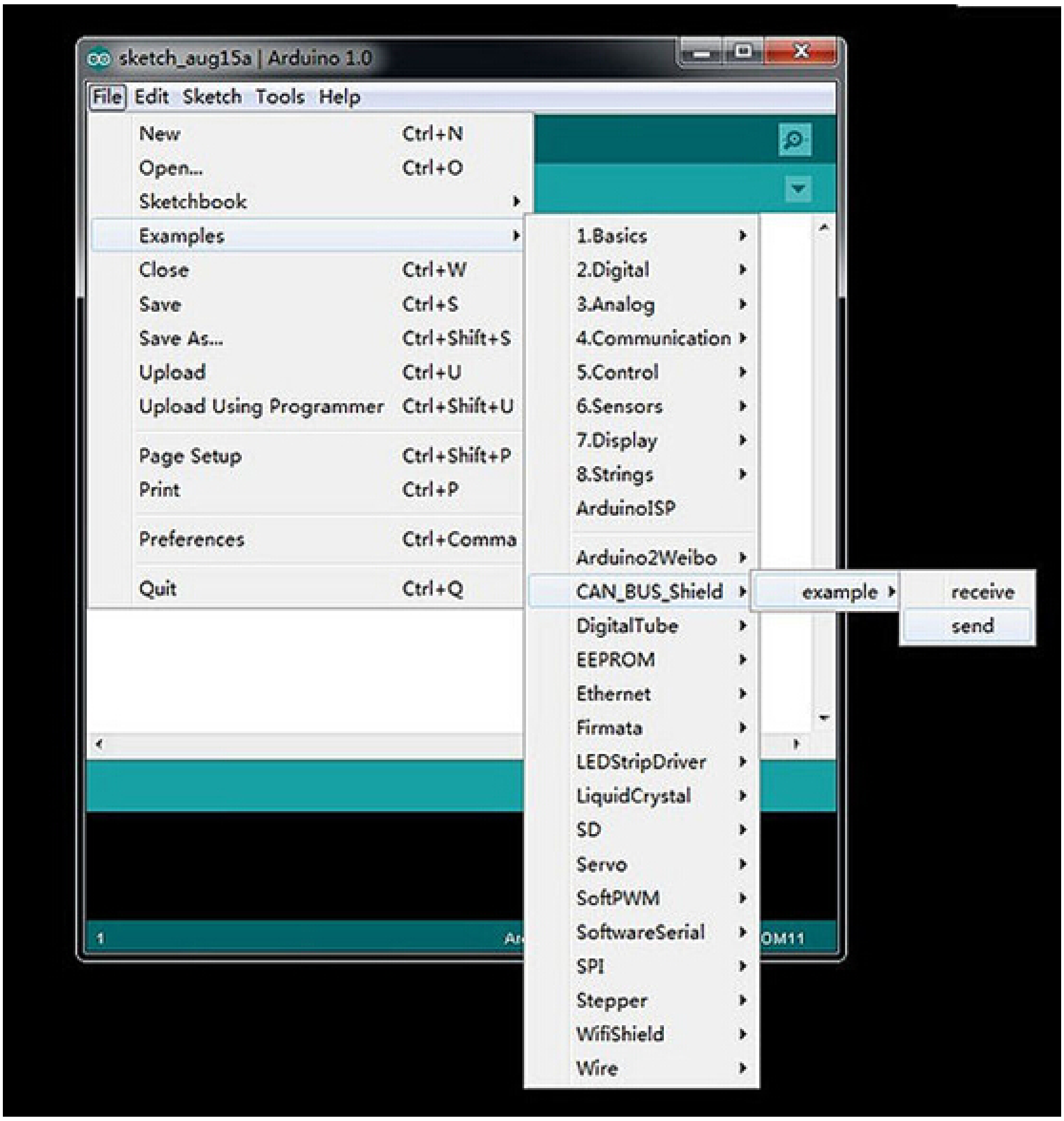
First, the CAN-BUS Shield into the UNO, then plug wiring diagram press.



Step 3

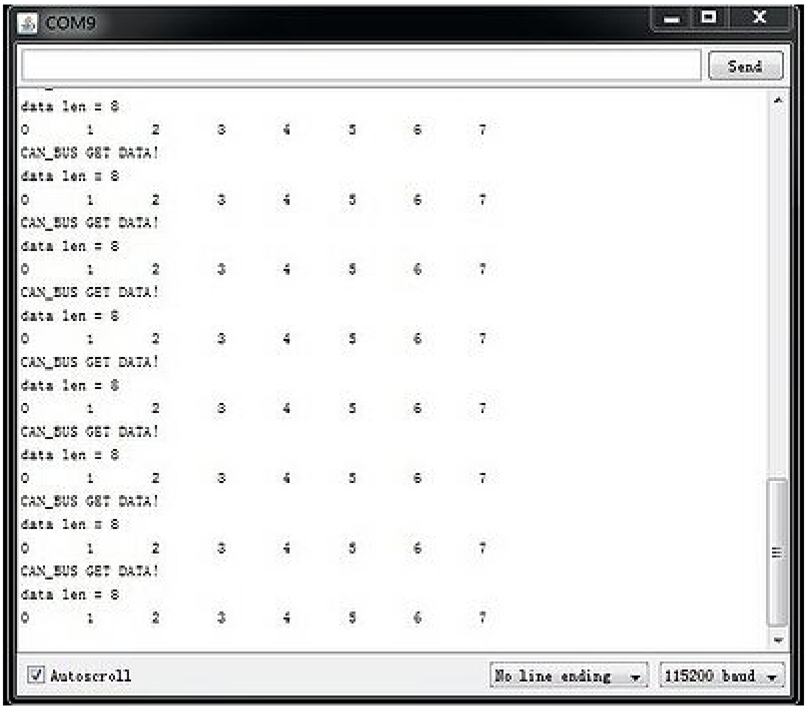
1. Download the CAN-BUS Source code and release it in the libraries file in the Arduino-1.0 program.: ..\arduino-1.0\libraries.

2. Open the Arduino-1.0, and you will find two examples: "receive" and "send". Open both of them, you should get two programming windows now



3. Upload two examples to two boards separately. Choose the board via the path: Tools -->Serial Port-->COMX. Note down which board is assigned as a "send" node and which board is assigned as a "receive" node.

4. Open the "Serial Monitor" on the "receive" COM, you will get message sent from the "send" node. Here we have the preset message "0 1 2 3 4 5 6 7" showing in the following picture



```
data len = 8
0	1	2	3	4	5	6	7
CAN_BUS GET DATA!
data len = 8
0	1	2	3	4	5	6	7
CAN_BUS GET DATA!
data len = 8
0	1	2	3	4	5	6	7
CAN_BUS GET DATA!
data len = 8
0	1	2	3	4	5	6	7
CAN_BUS GET DATA!
data len = 8
0	1	2	3	4	5	6	7
CAN_BUS GET DATA!
data len = 8
0	1	2	3	4	5	6	7
CAN_BUS GET DATA!
data len = 8
0	1	2	3	4	5	6	7
CAN_BUS GET DATA!
data len = 8
0	1	2	3	4	5	6	7
```

☑ Autoscroll   No line ending   115200 baud

Note: Note: Make clear of the transmit/receive modules when burning codes.

Programming

Includes important code snippet. Demo code like :

```
Demo code
{
// demo: CAN-BUS Shield, receive data
#include <mcp_can.h>
#include <SPI.h>

unsigned char Flag_Recv = 0;
unsigned char len = 0;
unsigned char buf[8];
char str[20];

void setup()
{
  CAN.begin(CAN_500KBPS);                       // init can bus : baudrate = 500k
  attachInterrupt(0, MCP2515_ISR, FALLING);     // start interrupt
  Serial.begin(115200);
}

void MCP2515_ISR()
{
    Flag_Recv = 1;
}

void loop()
{
    if(Flag_Recv)                           // check if get data
    {
      Flag_Recv = 0;                        // clear flag
      CAN.readMsgBuf(&len, buf);            // read data,  len: data length, buf: data buf
      Serial.println("CAN_BUS GET DATA!");
      Serial.print("data len = ");
      Serial.println(len);

      for(int i = 0; i<len; i++)            // print the data
      {
        Serial.print(buf[i]);Serial.print("\t");
      }
      Serial.println();
    }
}

/****************************************************************************************************
  END FILE
*****************************************************************************************************

}
```

Example

The projects and application examples.

Version Tracker

Revision	Descriptions	Release
v0.9b	Initial public release	date

Bug Tracker

Bug Tracker is the place you can publish any bugs you think you might have found during use. Please write down what you have to say, your answers will help us improve our products.

Additional Idea

The Additional Idea is the place to write your project ideas about this product, or other usages you've found. Or you can write them on Projects page.

How to buy

Click here to buy:

https://www.auselectronicsdirect.com.au/can-bus-

shield-for-arduino

See Also

Other related products and resources.

Licensing

This documentation is licensed under the Creative Commons Attribution-ShareAlike License 3.0 (http://creativecommons.org/licenses/by-sa/3.0/) Source code and libraries are licensed under GPL/LGPL (http://www.gnu.org/licenses/gpl.html) , see source code files for details.