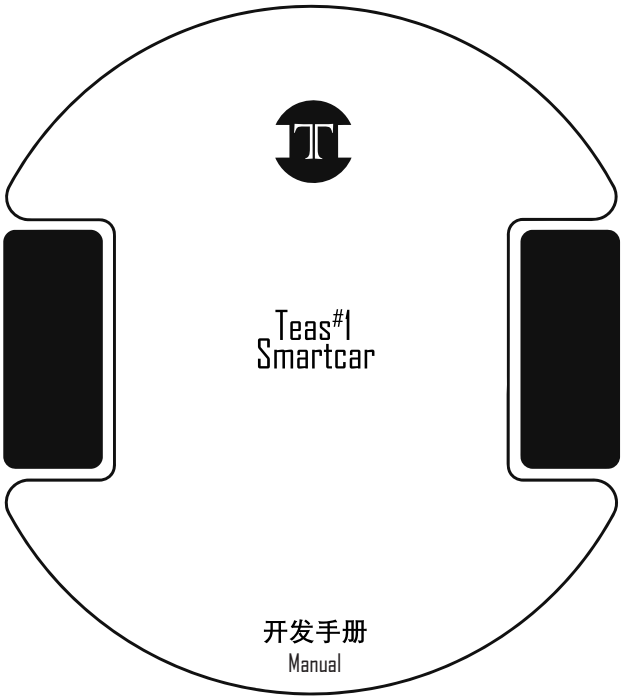
GREEN TEA SERIES



开发手册
Manual

TREEE TECH @ 2016

Let's Go

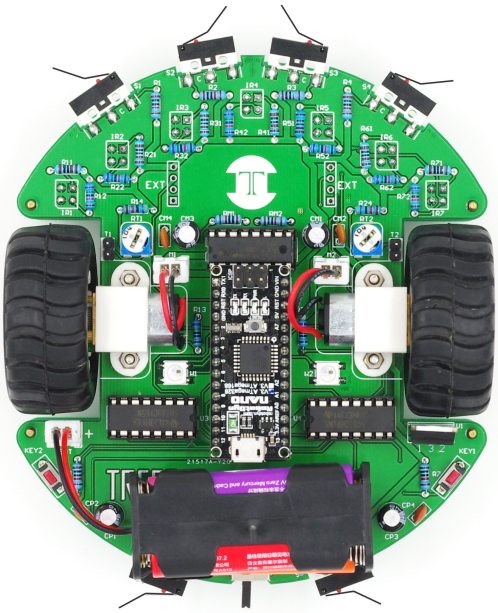


Figure 1: This is Teas $\sharp$1

Contents

Chapter 1

Introduction

1.1 Bill of Materials

All of the materials are list in Table 1.1. Please check if the materials in the box are complete. If there has a miss, please connect us.

Table 1.1: Bill of Materials

ID	Label / Location	Name	Model	Number
1	NA	Teas#1 PCB Main Board	NA	1
2	R11,R21,R31 ∼ R71	Resistor	1K	7
3	R12,R22,R32 ∼ R72	Resistor	75K	7
4	R1,R2 ∼ R8	Resistor	10K	8
5	RM1, RM2	Resistor	10K	2
6	R13, R23	Resistor	560	2
7	R14, R24	Resistor	47K	2
8	CM2, CM4	Capacitor	0.1uF	2
9	CP2, CP4	Capacitor	0.01uF	2
10	W1, W2	RGB Coloful LED	WS2812	2
11	U2,U3,U4	IC Socket	DIP16	3
12	ON—OFF	Power switch	NA	1
13	M1,M2,BAT	XH2.54 White Socket	2PIN	3
14	T1,T2	2.54MM Pin	2PIN	2
15	CM1,CM3,CP1,CP3	Capacitor	50V/47uF	4
16	KEY1, KEY2	Key	NA	2
17	S1, S2 ∼ S6	Collision switch	NA	6
18	RT1, RT2	Adj Resistor	100K Max	2
19	U1	DC Power IC	LM7805	1
20	PCB Middle	15PIN Pin	As MCU Socket	2
21	RIGHT / LEFT MOTOR	Motor Kit*	N20	2
22	On U2	Motor Driver IC	L293D	1
23	On U3, U4	Shift Register	74HC165	2
24	O1,O2	SMT IR Sensor	ITR8307	2
25	IR1, IR2 ∼ IR7	IR Sensor	ITR20001	7
26	MAGNET	Magnet Kit**	NA	1
27	On Magnet	AAA x 4 Battery Case	NA	1
28	On MCU Socket	Arduino Nano	Atmel Mega328p	1

* Motor Kit: Motor, Wheel, Cable for Motor, Motor Mount, M2 Screw x2, M2 Gasket x2

** Magnet KitUniversal WheelM2 Screw x 2, M2 Gasket x 4, Magnet

Note: In the process of taking components and welding, please pay attention to safety.

1.2 Resistor

A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. Resistors act to reduce current flow, and, at the same time, act to lower voltage levels within circuits. In electronic circuits, resistors are used to limit current flow, to adjust signal levels, bias active elements, and terminate transmission lines among other uses. High-power resistors, that can dissipate many watts of electrical power as heat, may be used as part of motor controls, in power distribution systems, or as test loads for generators. Fixed resistors have resistances that only change slightly with temperature, time or operating voltage. Variable resistors can be used to adjust circuit elements (such as a volume control or a lamp dimmer), or as sensing devices for heat, light, humidity, force, or chemical activity.

Resistor is represented by Rx. The behavior of an ideal resistor is dictated by the relationship specified by Ohm's law:

$$R = \frac{U}{I}[Unit : \Omega = \frac{V}{A}] \tag{1.1}$$

The resistors used in teas1 are 5 band ones, which have bands indicating the values or ratings of electronic components. The 10K resistor is shown in 1.1. Since decoding the 5 band resistor color code is difficult, we recommend to use multimeter for measurement.

Figure 1.1: $10K\Omega$ Resistor

1.3 Capacitor

A capacitor (originally known as a condenser) is a passive two-terminal electrical component used to store electrical energy temporarily in an electric field. The forms of practical capacitors vary widely, but all contain at least two electrical conductors (plates) separated by a dielectric (i.e. an insulator that can store energy by becoming polarized). The conductors can be thin films, foils or sintered beads of metal or conductive electrolyte, etc. The nonconducting dielectric acts to increase the capacitor's charge capacity. Materials commonly used as dielectrics include glass, ceramic, plastic film, air, vacuum, paper, mica, and oxide layers. Capacitors are widely used as parts of electrical circuits in many common electrical devices. Unlike a resistor, an ideal capacitor does not dissipate energy. Instead, a capacitor stores energy in the form of an electrostatic field between its plates.

The voltage across the capacitor is determined by the initial state of the capacitor and the charging and discharging time period. If the voltage across the capacitor is U_0 at $t = 0s$, then after time t, the voltage the capacitor will be:

$$U(t) = \int_0^t \frac{I(T)}{C} dT + U_0 \tag{1.2}$$

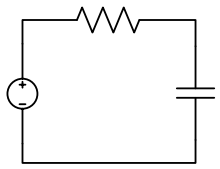


Figure 1.2: RC series connected circuit

If a capacitor is series connected with a resistor and a power supply with voltage V, the voltage across the capacitor will be:

$$U(t) = V(1 - e^{-\frac{t}{RC}}) \tag{1.3}$$

1.4 Diode

A diode is a two-terminal electronic component that conducts primarily in one direction (asymmetric conductance); it has low (ideally zero) resistance to the flow of current in one direction, and high (ideally infinite) resistance in the other. A semiconductor diode, the most common type today, is a crystalline piece of semiconductor material with a pn junction connected to two electrical terminals.

Different diodes have different threshold voltages (usually between 0.3V to 1V). After conducting, the diode is not absolutely equal to wire. There is a voltage drop across it, which approximately equals to the threshold voltage.

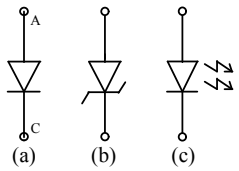


Figure 1.3: Symbols of: (a)Diode; (b)Zener Diode; (c)LED

A Zener diode is a diode which allows current to flow in the forward direction in the same manner as an ideal diode, but also permits it to flow in the reverse direction when the voltage is above a certain value known as the breakdown voltage.

A light emitting diode (LED) is a two-lead semiconductor light source. It is a pn junction diode, which emits light when activated. When a suitable voltage is applied to the leads, electrons are able to recombine with electron holes within the device, releasing energy in the form of photons. This effect is called electroluminescence, and the color of the light (corresponding to the energy of the photon) is determined by the energy band gap of the semiconductor.

1.5 Infrared Sensor

An IR sensor is a device which detects IR radiation falling on it. There are numerous types of IR sensors that are built and can be built depending on the application. Proximity sensors (Used in Touch Screen phones and Edge Avoiding Robots), contrast sensors (Used in Line Following Robots) and obstruction counters/sensors (Used for counting goods and in Burglar Alarms) are some examples, which use IR sensors.

An IR sensor is basically a device which consists of a pair of an IR LED and a photodiode which are collectively called a photo-coupler or an opto-coupler. The IR LED emits IR radiation, reception and/or intensity of reception of which by the photodiode dictates the output of the sensor. If the object is reflective, (White or some other light color), then most of the radiation will get reflected by it, and will get incident on the photodiode. For further understanding, please refer to the left part of the illustration below. If the object is non-reflective, (Black or some other dark color), then most of the radiation will get absorbed by it, and will not become incident on the photodiode. It is similar to there being no surface (object) at all, for the sensor, as in both the cases, it does not receive any radiation.

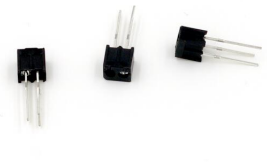


Figure 1.4: DIP IR Sensor

Figure 1.5: SMT IR Sensor for testing speed

Teas ♯1 uses reflective infrared sensor to detect the condition of ground as shown in Fig.1.4 and Fig.2.6. Principles and applications of reflective infrared sensor will be mentioned in Section 2.4.

1.6 Micro switch

A miniature snap-action switch, also trademarked and frequently known as a micro switch, is an electric switch that is actuated by very little physical force, through the use of a tipping-point mechanism, sometimes called an 'over-center' mechanism.

Switching happens reliably at specific and repeatable positions of the actuator, which is not necessarily true of other mechanisms. They are very common due to their low cost and durability, greater than 1 million cycles and up to 10 million cycles for heavy duty models. This durability is a natural consequence of the design.

The defining feature of micro switches is that a relatively small movement at the actuator button produces a relatively large movement at the electrical contacts, which occurs at high speed (regardless of the speed of actuation). Most successful designs also exhibit hysteresis, meaning that a small reversal of the actuator is insufficient to reverse the contacts; there must be a significant movement in the opposite direction. Both of these characteristics help to achieve a clean and reliable interruption to the switched circuit.

Figure 1.6: Micro switch

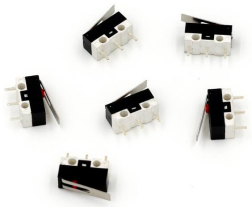


Figure 1.7: Collision switch

The switch shown in Fig.1.6 is one normal micro switch. It is used as operation key of Teas ♯1 . The switch shown in Fig.1.7 is also a kind of micro switch which is usually used to detect the impact situation. Teas ♯1 applies it to detect the collision.

1.7 IR Speed Sensor

The light source of the speed sensor is from the Infrared diode. When the reflected infrared is detected by the receiver, a step signal generates to get the rotating speed. Adding to the size of wheels, the speed of the car can be calculated.

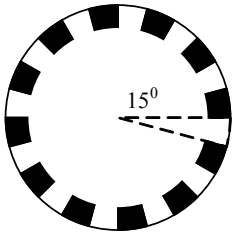


Figure 1.8: Schematic diagram of the speed measuring part of the wheel

The part used for speed measurement is shown in Fig. 1.8. The black part stands for the entity part which is opaque. In this manual, it is named velocity measure tooth(VMT). White part represents empty.

There are 12 small VMTs in one wheel. For each small tooth, it occupy $1/24$ of one circle of the wheel. For ever time the VMT passes the IR speed sensor, the voltage state will change one time. By measuring the speed change of voltage state, the seep of the wheel can be calculated.

1.8 RGB Colorful LED

It is known that the RGB color model is an additive color model in which red, green and blue light are added together in various ways to reproduce a broad array of colors. Therefore, at least three colors (RGB) of LED are necessary to generate more colors by adjusting the brightness by PWM wave. WS2812 consisting of LED and driver can simplify the process. WS2812 can control all the three colors with a data line. We will introduce how to use it in the later chapter.

1.9 PCB

A printed circuit board (PCB) mechanically supports and electrically connects electronic components using conductive tracks, pads and other features etched from copper sheets laminated onto a non-conductive substrate.

PCBs can be single sided (one copper layer), double sided (two copper layers) or multi-layer (outer and inner layers). Multi-layer PCBs allow for much higher component density. Conductors on different layers are connected with plated-through holes called vias. Advanced PCBs may contain components-capacitors, resistors or active devices embedded in the substrate.

PCBs are designed with dedicated layout software, such as Altium Designer, KiCad and Eagle CAD. Altium Designer, the upgraded version of Protel 99se conform to the common practice. KiCad is open source software but learned more difficultly.

1.10 Gearmotor

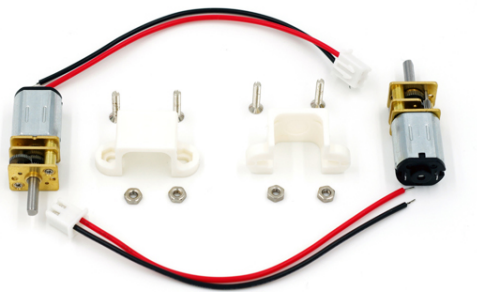


Figure 1.9: N20 Gearmotor

Gear motors are complete motive force systems consisting of an electric motor and a reduction gear train integrated into one easy to mount and configure package. This greatly reduces the complexity and cost of designing and constructing power tools, machines and appliances calling for high torque at relatively low shaft speed or RPM. Gear motors allow the use of economical low horsepower motors to

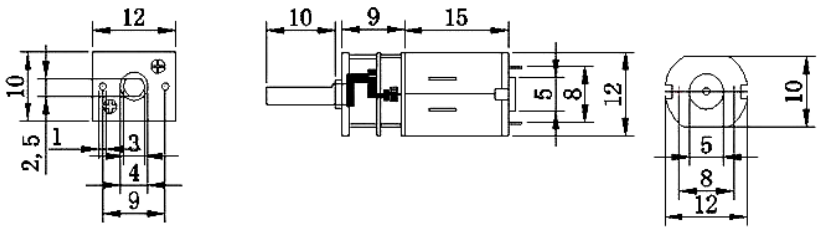


Figure 1.10: Package outline of N20 motor

provide great motive force at low speed such as in lifts, winches, medical tables, jacks and robotics. They can be large enough to lift a building or small enough to drive a tiny clock.

Theoretically, motors do not need to use gear to increase the force. But the driver should support enough power, which is a huge challenge of circuit design. Therefore, a gearbox is often used to increase the force.

1.11 Universal Wheel

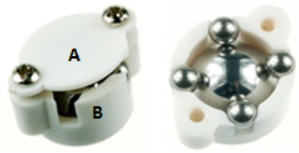


Figure 1.11: Universal Wheel

Teas 1 applies two dependent motor to drive two wheels. Therefore, a universal wheel is needed to keep balance. A universal wheel is composed by 4 small steel balls and 1 big steel ball with diameter 12 mm. The total height of the universal wheel is 1.5 cm. According to the using requirement, the total high can be adjusted by adding spacer. The shell is formed by two parts 'A' and 'B'. In case of losing steel balls, it is suggested to fix the two parts together by tape.

1.12 Arduino

Figure 1.12: Logos of Arduino and Genuino

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino

boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is open-source, and it is growing through the contributions of users worldwide.

1.13 STM32 microcontroller

STM32 is a family of 32-bit microcontroller integrated circuits by STMicroelectronics. The STM32 chips are cost-effective so that they are wildly used in industry design. Teas 1 uses the most wildly used STM32 F103 72 MHz Cortex-M3 core, 128 KB flash, 20 KB SRAM, external static memory interface. It also has extensive development reference as well. We specially designed a STM32 Nana board which is compatible with the function of the Arduino Nano pins and are improving the function library so that in STM32 users can operate in the way similar to Arduino language. Refer ¡***¿ for details.

Note: It is an optional part. Some of the product will not provide STM32.

1.14 Quiz

Problem 1 According to **??**, practice how to read the value of resistor and verify the value by using multimeter.

Problem 2 Practice how to use multimeter to measure voltage, current and resistance.

Problem 3 Search on the Internet and find that why the Tesla E-Car can have the performance as a super racing-car.

Chapter 2

Circuit Schematic

2.1 Diagram of Teas ♯1

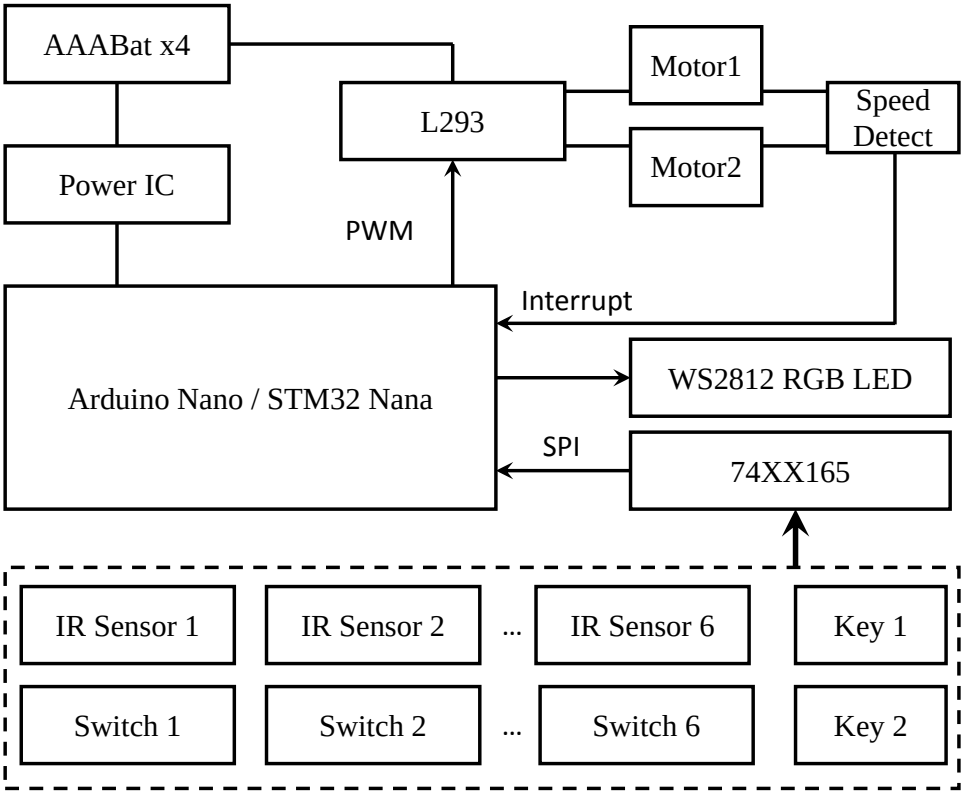


Figure 2.1: Teas ♯1 Diagram

2.2 Voltage Regulator

In any electronic system, voltage regulation could be the most important part. The main function of voltage regulation is to regulate the unstable or unsuitable input voltage to a suitable voltage with required current output. As different electronic components need different operating voltage, we should design a proper voltage regulator circuit to satisfy them.

The main components in the system of Teas ♯1 system are motors, Arduino Nano Micro Control Unit (MCU), STM32 microcontroller, Infrared (IR) sensors and some other ICs. The related voltage of Arduino control unit is +5V while that of STM microcontroller is +3.3V. Therefore, we have to consider how to make the two controllers work at the same time. The model of Arduino controller is Atmel AVR Mega 328P, which has high level threshold of 0.6VCC that is to say +3V. Hence, +3.3V signal can be detect by Arduino. On the other hand, STM32 controller only has small part of pins are compatible with +5V signal. Thus, most components use the power supply of +3.3V.

The related voltages are DC +3.3V, +5V and the battery voltage (for motors). As the motor is a power output type component, big current is required to drive it. But the voltage regulator IC sometimes cannot support so large current. Therefore, generally the motor is driven by the power source through a motor driver circuit or IC. If the close-loop motor speed control is used, we can also regard the motor control method and driver circuit as a kind of high power voltage regulator. Other ICs and sensors need very stable +3.3V voltage while Arduino needs +5V voltage. Otherwise, the MCU will reboot frequently and the sensor data will not be accurate.

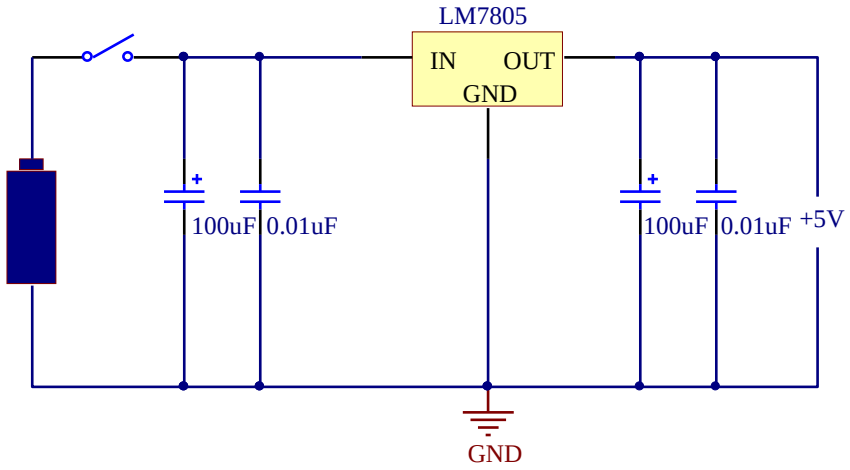


Figure 2.2: A voltage regulator design using LM7805 IC

A very mature power IC - LM7805 is applied to regulate the voltage of 5.5V in Teas ♯1 as shown in figure 2.1. LM7805 is a typical voltage regulator IC with three pins which are power source voltage input, +5V voltage output and the common ground (GND) respectively. In order to make the output voltage more stable, capacitor-based filters are applied at the power source input and +5V voltage output. A filter consists of a small capacitor and a big capacitor to filter the high-frequency noise and low-frequency noise respectively. In order to achieve stable voltage of +3.3V, we use the LDO regulator LM(AMS)1117-3.3 adding to some filters. LM1117 is also a 3-pin voltage regulator. It has similar connection as LM7805. Arduino Nano has its built-in LM1117 to output 3.3V voltage. And we have added LM1117 in the design of STM32 Nana. Therefore, there is no need to consider the 3.3V voltage regulator on the PCB.

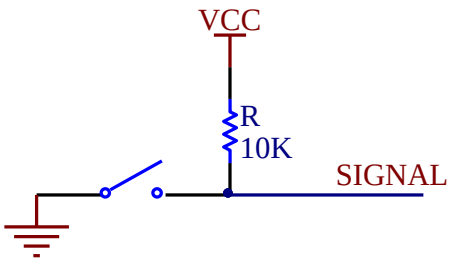
Currently, there are many high efficiency and expensive voltage and current regulator ICs. The two main types are LDO regulator and switching mode regulator. Generally, LDO regulator has more stable voltage without ripple but low efficiency, while switching power supply regulator is opposite. When we

choose an IC, the first factor is the requirement of voltage and current. The output voltage needs to be the same as ICs requirement and the current should be larger than the total current required by all the components. The capacitor selection can refer the datasheet of the IC. It is worth mentioning that big capacitor is good at make the voltage stable, but it takes longer for the voltage to achieve stability which will impact on the start of MCU.

DC switching mode power supply applies PWM duty cycle to adjust the output power and use the negative feedback to keep the voltage stable. Generally, the DC switching mode power supplies have two types which are boost and buck. The peripheral circuit of DC power IC usually requires capacitances and inductances designed by switching frequency and the load, and voltage divider feedback resistor. Generally, we can design a suitable voltage regulation circuit of switching mode power supply. Teas ♯1 does not have a switching mode power supply.

2.3 Circuits of Collision Detection and Keys

An important function of Teas ♯1 is to avoid obstruction. So an obstruction detection circuit is designed to sense whether the smartcar meets the obstruction. A contact-type collision sensing circuit is applied by using some long-arm switch. The advantage of using contact-type method is that the smartcar can push some very light objects instead of avoiding them. Six switchs are implemented around the smartcar: four in front and two at the back, which can detect many directions.



开关状态	输出信号
闭合	0
打开	1

Figure 2.3: The switch circuit and the state of output signal.

The principle of switch is that when a switch is pressed, two of its 3 pins will be connected. One of them is connected to GND, while the other one is connected to the MCU with a pull-up register. The reason of using pull-up register is to make the signal stable. When the switch is not pressed, the signal output connected to the MCU is impending, and the input pin of MCU is high-impedance state. Therefore, a 10k ohm register is often used as a pull-up register for MCU.

In addition to the six collision switchs, two keys (or buttons) is designed for user to develop multiple functions, which located at the back of Teas ♯1 smartcar. Its principle is the same as collision switchs.

2.4 Circuits of IR Sensor and Speed Detection

Teas ♯1 could detect whether the ground color is dark or light, which could achieve the black line tracking and avoiding falling, etc. An integrated IR sensor with an IR LED emitter and IR receiver is exploited as

the basic component with a simple circuit as shown in 1.4.

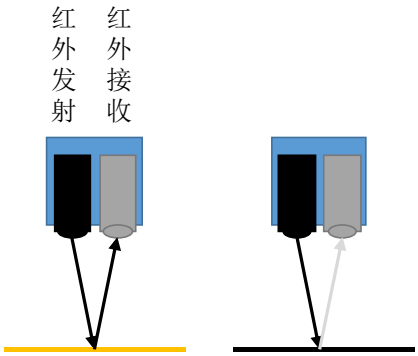


Figure 2.4: The black line detection principle using the integrated IR sensor.

The IR receiver is a IR photistor, which is similar to a transistor. The difference is that the base is replaced by photosensitive materials. When the light meet the base, the current can go through the photistor, which is similar to a transistor. Fig.2.5 shows the schematic of IR sensor, which is very simple.

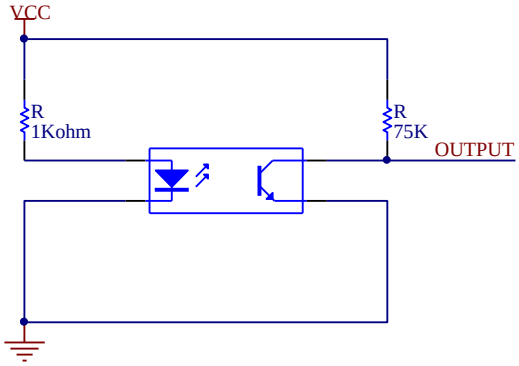


Figure 2.5: The circuit schematic of IR sensor.

In Teas ♯1 , the motion of smartcar is driven by two independent motors. With many realistic reason, these two motor cannot be total same. With the same voltage and current, their speeds might be different. Therefore, the speed close loop control has to be exploited to address this phenomenon. In hardware part, the speed sensing is applied by using SMT (surface mount technology) IR sensor. As there are 12 white bars in the tire, when the tire rotates, the small IR sensor can detect different colors and generate pulse. The MCU gets speed by sampling the pulse signal.

2.5 Digital Signal Sampling with Shift Register

In Teas ♯1 there are 6 impact switchers, 6 IR sensor and 2 keys, each of which requires a digital input pin on the MCU. But there are only 13 digital IO pins in Arduino Nano. In addition, these signals do not require very high speed sampling. Therefore, it is considered to use a shift register, a digital IC that transform parallel signal to serial signal, so that the MCU only needs to read the serial signal of 3 or 4

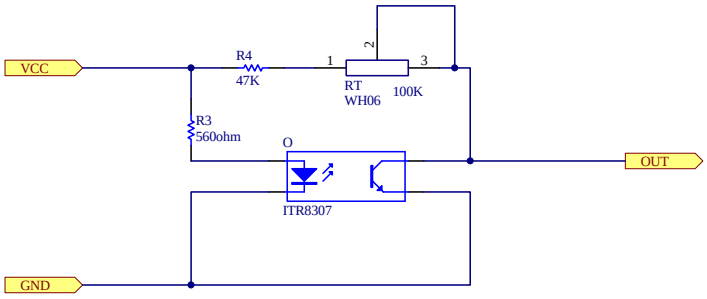


Figure 2.6: The circuit schematic of IR speed sensor.

pins. There are two types of shift register, parallel to serial one and serial to parallel one. A typical IC of parallel to serial shift register is 74XX165 (XX means HC or LS) and a typical IC of serial to parallel shift register is 74XX595. Teas ♯1 applies two 74HC165 together to get 14 parallel signals of switchers and sensors. 74HC165 is an 8-bit serial or parallel-in / serial-out shift register. The pin configuration of 74HC165 is shown in Fig.2.7 and pin description is in Table 2.1.

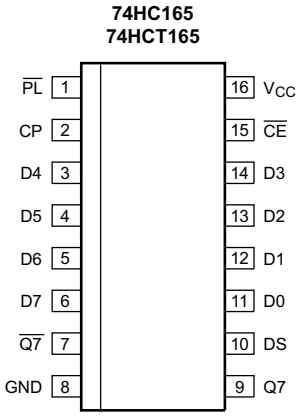


Figure 2.7: 74HC165 IC diagram.

In the use of 74XX165, the statement of all the pins should be stored in the shift register by setting the PL low firstly. Then binary data in it is outputted in the form of clock signal (LOW-to-HIGH edge-triggered) by the Q7. When cascading a few 74XX165, the Q7 of former register is connected to the DS of the next one. Then the PL and the CP of all the registers should be connected in parallel. The data is out put through the last Q7. Finally, the MCU can read the serial signal by sending 16 clock pulse, and get the output from Q7. The timing diagram is shown in Fig.2.8.

2.6 Motor Driver

The function of motor driver is to control the speed and direction of motor. The motor used in Teas ♯1 is DC brush motor, which is simple to control. In addition, there are many different types of motors, such as DC brushless motor, AC motor, and stepper motor, etc.

The control strategy of DC brush motor is relatively simple. When the motor connects to the DC

Table 2.1: 74HC165 Introduction

Name	**Pin**	**Function**
$\overline{PL}$	1	Lock the data (low is active)
CP	2	Clock input (rising edge active)
$\overline{Q7}$	7	The opposition of Q7 output
GND	8	GND
Q7	9	Q7 output
DS	10	Serial input
D0-D7	11,12,13,14,3,4,5,6	Parallel input (Dn)
$\overline{CE}$	15	Clock enable
Vcc	16	VCC

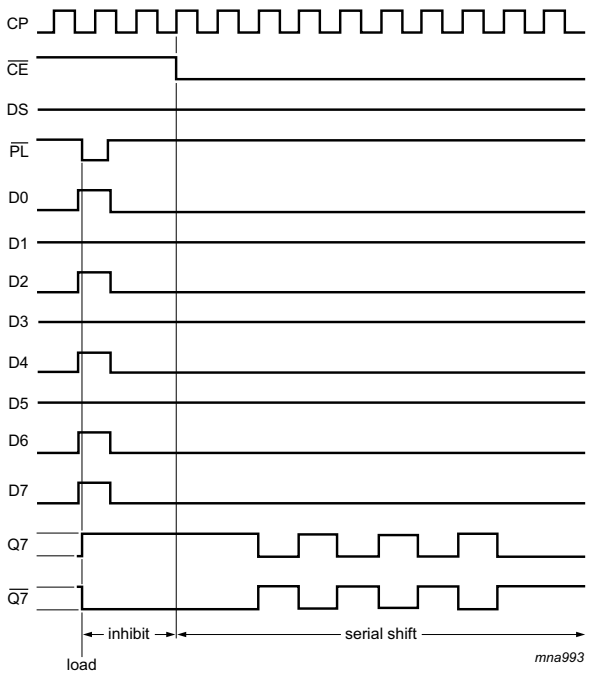


Figure 2.8: 74HC165 timing diagram.

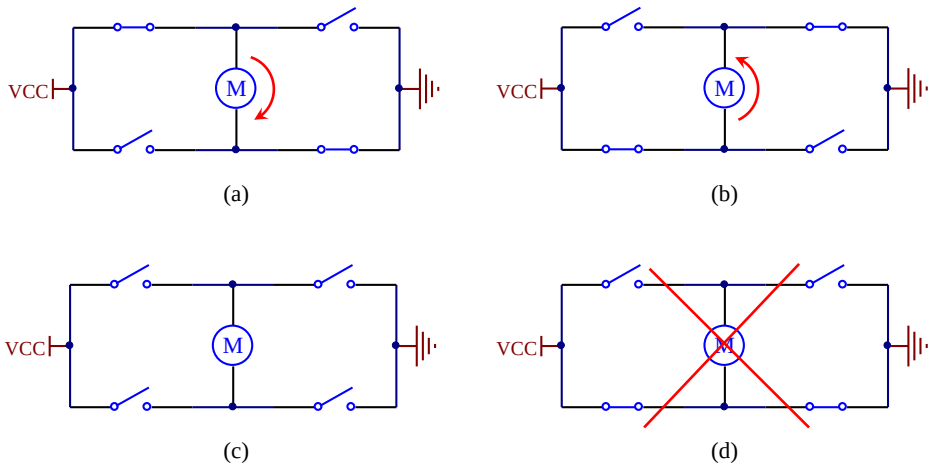


Figure 2.9: The diagram of full H-bridge motor driver control strategy.

power source, it can rotate, while when the we change the direction of power source, the rotation direction will change. But we cannot change the power source during the smartcar is moving. So we need to design a full H-bridge motor driver to control the motor. The basic principle of motor control strategy is shown in Fig.2.9. In (a), assume that the motor is rotating with clockwise. Then in (b), the motor will be rotating with anti-clockwise. In (c), the motor will lose power. The condition (d) must be forbidden as it is short circuit. In realistic case, the mechanical switchs cannot be used because the PWM frequency is very high. So the electronic switchs should be used. Transistors, MOSFETs and IGBT are generally used as electronic switchs.

We control the PWM duty cycle to control the speed of motor as shown in Fig.2.10. The bigger duty cycle can release more energy, which will make the motor faster. But it is difficult to control the accurate speed only using PWM duty cycle. If the load changes, the same duty cycle cannot drive the motor as the same speed. So close loop control has to be used to adjust the speed in real time. In order to get the rotation rate of the motor, speed sensor needs to be used, which is introduced before. When the speed is known, the PWM duty cycle will increase if the speed is lower than the expected speed, while the PWM duty cycle will decrease if the speed is higher than the expected speed. This might be the simplest way to control the speed of motor. A better way is PID control algorithm, which you can find many papers about this on IEEE.

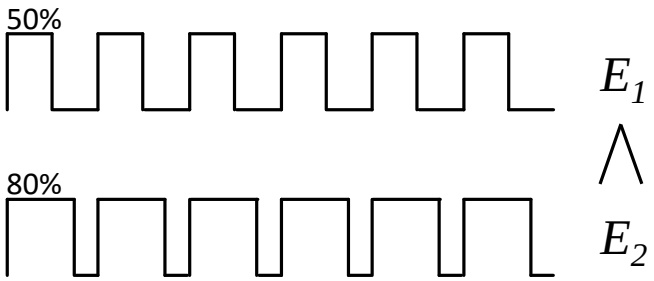


Figure 2.10: Power control using PWM wave.

In Teas ♯1 , the motor does not require a very high current. So the motor driver IC - L293 is used. Transistors are integrated in L293 as electronic switchs, which includes 2 groups of full H-bridge. It can control two motors, which is very suitable for Teas ♯1 .

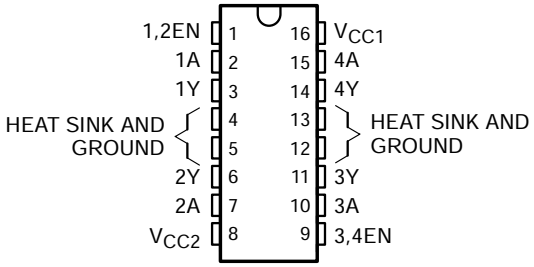
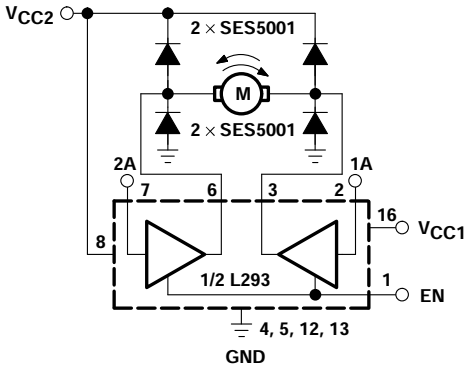


Figure 2.11: L293/L293D Pin Map.

Fig.2.11 introduces the pin map of L293 (L293D is a diode inside L293 version). VCC1 is connected to +5V, while VCC2 is connected to battery for driving motors. GROND is connected to GND. All the

pin started with A is PWM input, while Y means it should connect to motors. Fig.2.12 introduces an example of L293 based DC motor driver.



EN	1A	2A	FUNCTION
H	L	H	Turn right
H	H	L	Turn left
H	L	L	Fast motor stop
H	H	H	Fast motor stop
L	X	X	Fast motor stop

L = low, H = high, X = don't care

Figure 2.12: An example of DC motor based on L293.

Chapter 3

Installation Guide

For Teas ♯1 most components are SIP components. Only two SMT components are contained. Small phillips screwdriver will be used to fix mechanical components. While considering screwdriver is a very normal tool in laboratory, it is not provided in Teas ♯1 . In this chapter, the method of how to use solder iron, some attention tips and solder order will be introduced.

Safety is the most important thing. The power of Teas ♯1 is supplied by 4 series connected batteries, and the highest supply voltage should not be higher than $6V$. Since there are no hug capacitive components contained in Teas ♯1 , in theory, it is not possible to generate high voltage. In soldering process, user should be careful not to be burned by the high temperature iron. Develop good habits of using solder iron is an effective solution to prevent injuries. Do not use hand to try if the iron tip is heated or not. The temperature can be tested by using solder stick or wet sponge.

Attention trauma when using screws and scissors. Protective gloves, goggles and other safety equipment are very necessary. The position of the solder station is important. For a right-handed user, the solder station is suggested to be put on the right side. When taking the solder iron, the cable is not easy to be twined together. The cable spoil situation is not easy to happen. If the cable is spoiled, accident of electric shock may be leaded.

Besides the safety consciousness and habits, the necessary safety protection measures are also important. Laboratories should be equipped with the necessary emergency treatment chemicals. The fire protection equipment around should be clearly known. The use of the fire extinguisher should be familiarized.

3.1 DIP Soldering

Before soldering the solder tip should be kept clean. For the fist step of soldering , the connection point between the resistor lead and the bonding pad should be heated by solder tip. Be careful to heat two components at the same time.

After 1 to 2 seconds, put the solder stick to the solder point. Please do not put the solder stick to the solder tip directly, but to the intersection point at all components (the solder point) instead!

After enough solder melt, the solder stick should be removed from the point of contact at 45 degrees angle. Then removing the solder tip from the solder point with the same angle (45°). Finally, we can find a solid solder joint, and the redundant resistor lead can be cut down.

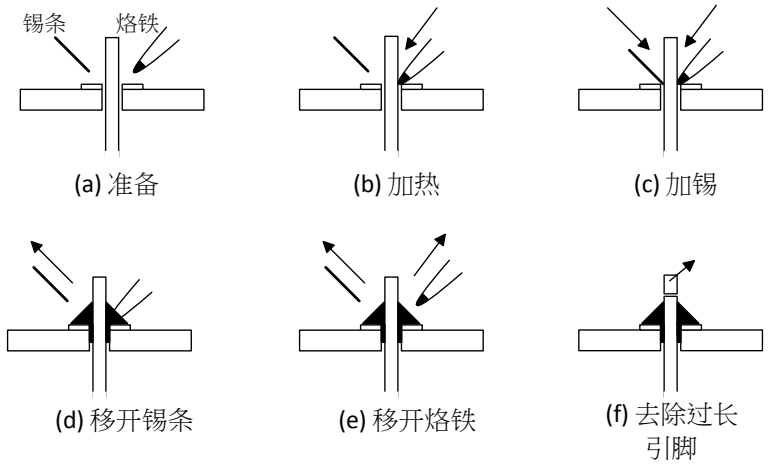


Figure 3.1: DIP Soldering Process

3.2 SMT Soldering

SMT (surface mount technology) parts have many advantages over their older through hole ancestors, namely: smaller size, lower cost manufacturing costs, and less likelihood of obsolescence.

Two terminal devices such as resistors and caps are generally the easiest parts to start out with. Just place the part onto its location on the PCB, heat the joint between the terminals on the part and the PCB while touch it with solder, and the solder will flow. After one side has been soldered quickly solder the opposite side in the same way. Surface tension of the liquid solder should center the part. Use the solder tip and the solder strand to push the part into place. If you get too much solder on the part, just use some solder wick to suck up any extra.

Soldering SOICs is not much different. Just line up the part. Tack a single lead down, and and push the part so the pins are aligned with the pads. Then solder each of the other pins. If solder bridges across any of the pins, it isn't a big deal. Just wipe away the solder with solder wick.

Most people are really intimated by the big quad flat pack (QFP) parts that have a hundred pins or more. However, these are just as easy to solder. People mistakenly think that they need to solder each pin individually without causing any solder shorts. In reality the approach is to tack it into position, and to then cover it with solder, ignoring any shorts, since these can easily be removed with solder wick.

3.3 Tips of Teas ♯1 Installation

To make sure the Teas ♯1 can be installed successfully, please read the following tips carefully.

Safety first: In the process of taking components and welding, please pay attention to safety, and be careful of scratches and burns.

Diodes: In the process of soldering, pay attention to the direction of the diodes. The white line on the diode should be corresponding to the white line on the PCB board.

Electrolytic capacitor: Attention the direction, the sign '+' on the board should be corresponding to the positive side of the Electrolytic capacitor.

IC Socket: The arc opening directions of the IC Socket and the sign on the PCB board should be identical.

Impact switch: The long arm stretching direction of the impact switch should be identical with the sign on the back of the PCB board.

IR sensor: Watch the infrared sensor from the up side, it can be seen that the sensor is not rectangular, but with a lack of angle. Make the position of the angle be same. Moreover, the infrared sensor should be soldered on the back of the PCB.

SMT IR sensor: Keep the position of the lack of angle identical. Attention the soldering time should be not too long.

Motor: Attention the positive side of the motor is signed. The red wire of the motor cable should be soldered with the positive side of the motor. In the process of soldering the motor socket, the gap of the socket should toward astern or toward the motor. If the motor cable is not soldered with recommended, the rotating direction can be adjusted through changing the code.

Universal wheel: In case of losing steel balls, before the screws are remove, the two shell are suggested to be fix together by using adhesive tape.

Battery box: The original wire of the battery box is very long. According to the distance requirement, the wire can be cut off partly.

Arduino Nano: Before using the MCU Arduino Nano , the needles should be soldered firstly (if it is not soldered). The 2x3 needle (Fig.3.2) is used to connect with ASP loader.

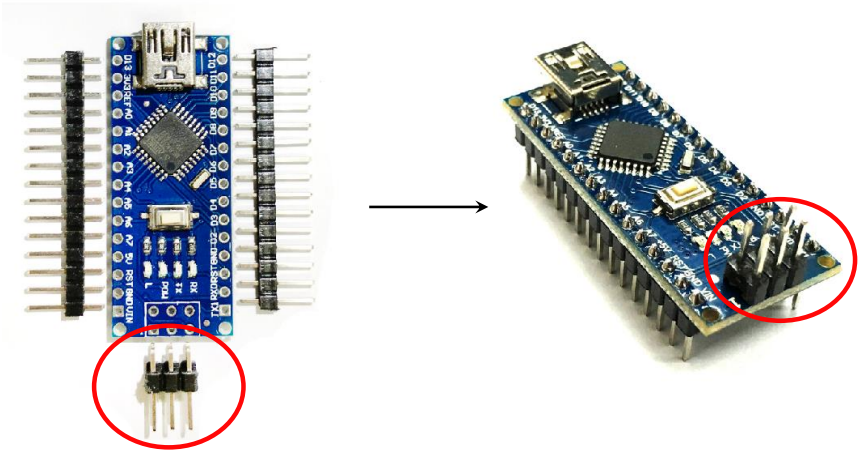


Figure 3.2: Arduino Nano Attention tips of soldering

Magnet: The main function of the magnet is used to absorb the battery and fix the battery box. For aesthetic consideration, fix the batteries by using cable tie is not applied. Fix magnet using the M2 screw together with universal wheel.

Sockets: The sockets with short legs is used to be soldered on the PCB board. The long leg sockets should be soldered on the expansion board.

3.4 Recommended Order of Installation

In the process of soldering PCB, following reasonable soldering order can increase soldering efficiency. The base principle is that starting the components from low to high and from the center to surround. Click here to watch the recommended soldering order .

Chapter 4

Arduino Programming

Arudino is an Almel family MCU based simple embedded system development platform with very simple open source IDE software. In their family, the most well-known Arduino MCU is UNO, which means ONE in Italian. It used AVR ATmega328P as the MCU core with 16MHz CPU frequency and 5V input voltage. It can meet many requirements of beginners. Besides, there are many powerful versions of Arduino, which can be accessed from their website: `https://www.arduino.cc`.

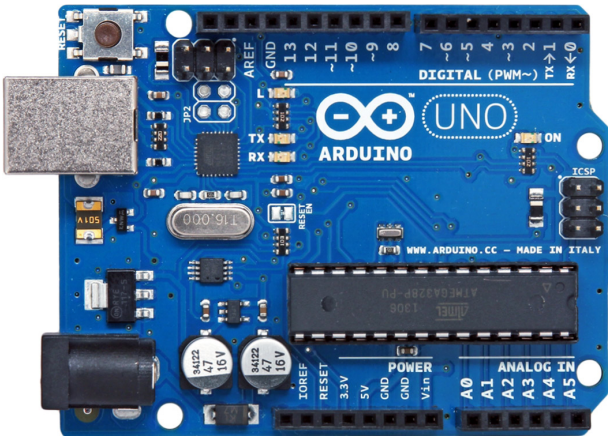


Figure 4.1: Arduino Uno Development Board.

In my view, the most important reason of the success of Arduino UNO is that the pin arrangement is asymmetry, which defines the special shape of its extension shields. Also the size of Arudino UNO is very suitable for many projects. The IDE software is simple enough, which reduce many difficulties of MCU development. The syntax is redesigned from C++, but it is much easier than C++. Many Arduino programs can be loaded into your directly, which make many beginners enter this field.

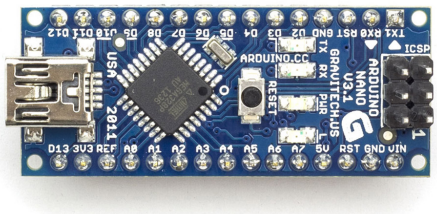


Figure 4.2: Arduino Nano Development Board.

However, Arduino Uno is too big for smartcar model. So we used a small size version - Arduino Nano as shown in Fig.4.2. There are 8 analog input pins and 14 digital IO, including 1 UART (D0, D1), 6 PWM pins (D3,D5,D6,D9,D10,D11), SPI port (D10 D13). These functions can meet the requirement of Teas ♯1 .

4.1 Arduino IDE

There are two main ways to load program from PC to Arduino. One is using Arduino IDE with USB, while the other is using AVR Studio with ASP loader. The simpler way is suing Arduino IDE, which is also the recommended way in this manual. If you are interested in the second way, please search the related materials on Internet.

It is very easy to obtain Arduino IDE, pleae visit HERE to download the last version. The Arduino IDE can support OS X, Windows and Linux. The current version is 1.6.7.

The language used by Arduino is very similar to C++, which also supports Object-Oriented Programming. All the libraries are developed using C++. Here we assume you learned C programming, and basic analog and digital circuit knowledge.

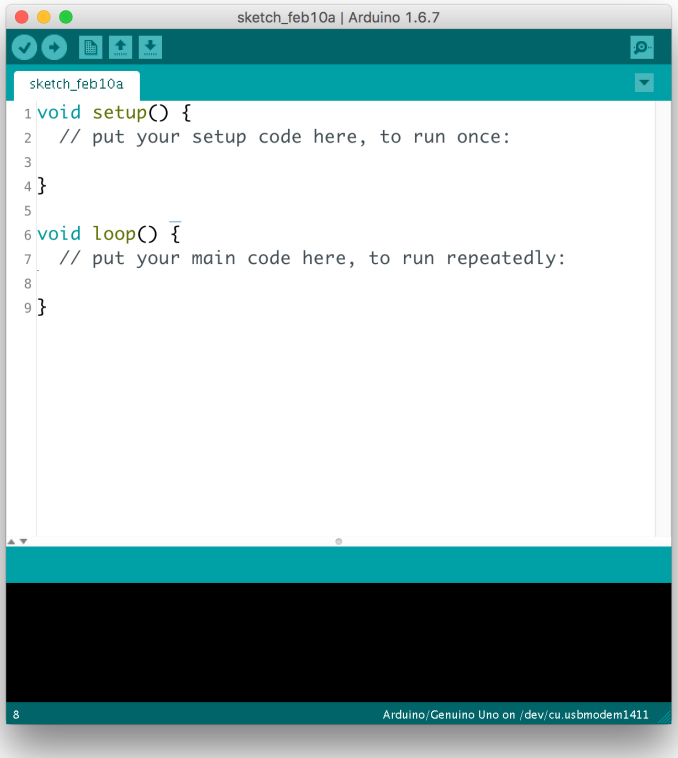


Figure 4.3: Arduino IDE GUI (OS X version)

As Fig.4.3 shown, when we open the Arduino IDE, two functions are built automatically, which are setup() and loop(). If you save the project, the extension name should be ino. Let's explain these two functions:

setup(): The setup() function is called when a sketch starts. Use it to initialize variables, pin modes,

start using libraries, etc. The setup function will only run once, after each powerup or reset of the Arduino board.

loop(): After creating a setup() function, which initializes and sets the initial values, the loop() function does precisely what its name suggests, and loops consecutively, allowing your program to change and respond. Use it to actively control the Arduino board.

For other functions and syntaxes, there are very similar to C++. The standard C libraries, such as strnig.h and stdio.h can also be included in Arduino project. In the following parts, some Arduino special functions, including digital IO operation, analog signal sampling, timing control, communication, etc, are introduced respectively.

4.2 Arduino Program Loading

When we are learning C/C++ languages, we known that in Windows Visual Studio, after finishing the code, we just press RUN and the code could become exe file for execution. Or in Linux or OS X, gcc is used as complier to make codes into executable files. But it is different for MCUs. We write MCU codes on PC and compile them, but they can only be executed on MCU.

As you expect, Arduino IDE make all the steps very simple. The USB port is used to load program. For Arduino UNO, the USB driver is included in the installation folder (driver folder). For Arduino Nano, the USB driver can be accessed from our website: Click Here to Download. Currently, OS X EL Capitan, Windows 7 and 10 are tested without any problem. It is worth mentioning that if you find that your Nano cannot be detected by Windows, please make sure that you use the genuine operating system.

4.3 Arduino Digital IO

Digital IO is the most important function in MCU. Output pins can control other peripherals with low and high level voltages. Input pins can sample the digital signal from outside. Arduino make IO operation very simple. Let's begin:

IO means input and output. So the direction of pin can be modified by pinMode() function:

Syntax:	pinMode(pin, mode);
Function:	Define the direction of a pin.
Arguments:	pin: the number of the pin whose mode you wish to set, in Arduino Nano , IO numbers are 0-13. mode: direction and mode, including INPUT, INPUT_PULLUP and OUTPUT.

For pins with OUTPUT, they can output low (0V) or high level (5V in Nano) voltage. Note that the output power is very limited, which is only 40mA. It can only light a LED, but not for driving motor and relay directly. Let's see how to operate:

Syntax:	digitalWrite(pin, value);
Function:	Write a HIGH or a LOW value to a digital pin.
Arguments:	pin: the pin number value: HIGH or LOW

In terms of pins with INPUT or INPUT_PULLUP, they can get the digital signal. The voltage near 5V can be detected as 1, while around 0V can be detected as 0. When a pin is not connected to anything, it is undetermined. So if the input signals are OC, OD signal, or switcher, a pull-up resister should be connected to input pin. The pull-up resistor can be an real resistor, or the MCU integrated resister can be used. The INPUT_PULLUP mode means the pin is pulled-up by the integrated resister. How to read from input pin is introduced as follow:

Syntax:	digitalRead(pin);
Function:	Reads the value from a specified digital pin, either HIGH or LOW.
Arguments:	pin: the number of the digital pin you want to read.
Return:	HIGH or LOW

Here is an example to practice digital IO operation.

Example: Write a code to control a LED by pressing a key. Press once, the LED is ON. Then press again, the LED is OFF.

Tips: In Aruino Nano , a LED is connected to pin D13 already, which is ON with HIGH. We set pin D12 as input pull-up mode for a key. Please read the following code carefully.

```
#define LED 13
#define KEY 12

void setup() {
  //Set the direction of LED as OUTPUT
  pinMode(LED, OUTPUT);
  digitalWrite(LED, LOW);
  //Set the direction of KEY as INPUT with pull up
  pinMode(KEY, INPUT_PULLUP);
}

void loop() {
  if (!digitalRead(KEY)){
    //When you press the button, it may shake, so
    //check it again after a while to ensure that it is pressed
    delay(100);
    if (!digitalRead(KEY)){
      //digitalRead can also read the state of OUTPUT pin
      digitalWrite(LED, !digitalRead(LED));
      //Wait for button release
      while (!digitalRead(KEY));
    }
  }
}
```

This code seems very simple, but please understand how to program on MCU. The first step is setup, which set the initial mode and information for each pin and component. Then, in the loop, the key is

scanned frequently. As the speed of MCU is much faster than the speed pressed by people, the MCU can get the key state under a relative real time condition. Note that in line 13, 16 and 17, we determine the key twice in order to reduce the vibration of the mechanical component. You can try this code with these lines and see what will happen. In line 21, we use a while loop to determine the key is released. But it is not very efficient. You can try to improve this part.

4.4 Arduino Serial Communication (UART)

The serial communication is very popular in computer. UART (universal asynchronous receiver-transmitter) is very commonly used between PC and MCU. In Arduino, the UART communication is also very simple by using some functions.

Firstly, the UART should start:

Syntax:	`Serial.begin(speed);`
Function:	Set the speed of UART and start
Arguments:	speed: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, or 115200

Then, if we want to receive the data, we need to determine the data is available or not:

Syntax:	`Serial.available();`
Function:	Get the number of bytes (characters) available for reading from the serial port.
Arguments:	(NA)
Return:	the number of bytes available to read

To read a byte or read some bytes:

Syntax:	`Serial.read();`
Function:	Read a byte from serial port
Arguments:	(NA)
Return:	the first byte of incoming serial data available (or -1 if no data is available)

Syntax:	`Serial.readBytes(buffer, length);`
Function:	reads characters from the serial port into a buffer
Arguments:	buffer: the buffer to store the bytes in (char[] or byte[])
Return:	byte

Syntax:	Serial.print(val, [format]); Serial.println(val, [format]);
Function:	Prints data to the serial port as human-readable ASCII text
Arguments:	val: the value to print - any data type format: specifies the number base (for integral data types) or number of decimal places (for floating point types)
Return:	number of bytes written

The printf() in C language can be also used in Arduino. The following example introduces a piratical way to user UART in Arduino.

Example: Use PC to control the LED (D13). Send “LED0”, the LED is off and send back “LED is Off” to PC, while send “LED1”, the LED is On and send back “LED is On”. The test result is shown in 4.4.

```
#include <stdio.h>

#define LED 13

String inputString = "";         // a string to hold incoming data
boolean stringComplete = false;  // whether the string is complete


// Use printf in Arudino
int serial_putc( char c, struct __file * ){
  Serial.write(c);
  return c;
}

void printf_begin(void){
  fdevopen(&serial_putc, 0);
}

void setup() {
  // initialize serial:
  Serial.begin(9600);
  // reserve 200 bytes for the inputString:
  inputString.reserve(200);
  // LED
  pinMode(LED, OUTPUT);
  printf_begin();
}

void loop() {
  // print the string when a newline arrives:
  int num = 0;
  if (stringComplete) {
    sscanf(inputString.c_str(), "LED%d",&num);
    if (num == 1){
      printf("The␣LED␣is␣On.\n");
      digitalWrite(LED, HIGH);
    }
```

```
    else if (num == 0){
      printf("The␣LED␣is␣Off.\n");
      digitalWrite(LED, LOW);
    }
    // clear the string:
    inputString = "";
    stringComplete = false;
  }
}

void serialEvent() {
  while (Serial.available()) {
    // get the new byte:
    char inChar = (char)Serial.read();
    // add it to the inputString:
    inputString += inChar;
    // if the incoming character is a newline, set a flag
    // so the main loop can do something about it:
    if (inChar == '\n') {
      stringComplete = true;
    }
  }
}
```

Figure 4.4: Using Terminal tool in Arduino IDE, and test the code

This example can be modified to used in many cases. Please read the code carefully and understand each part.

4.5 Arduino Analog to Digital Converter

The Atmega controllers used for the Arduino contain an onboard 6 channel analog-to-digital (A/D) converter. The converter has 10 bit resolution, returning integers from 0 to 1023. While the main function of the analog pins for most Arduino users is to read analog sensors, the analog pins also have all the functionality of general purpose input/output (GPIO) pins (the same as digital pins 0 - 13).

Consequently, if a user needs more general purpose input output pins, and all the analog pins are not in use, the analog pins may be used for GPIO.

As many sensors output only analog signal, the ADC can be used to convert the analog signal to digital data. The digital data is the input voltage divide the reference voltage and than times 1024. So the reference voltage setting is very important for ADC:

Syntax:	analogReference(type);
Function:	Set the reference voltage type
Arguments:	type: Types: DEFAULT: 5V INTERNAL: 1.1V EXTERNAL: From outside

To read the value from ADC is also very simple:

Syntax:	analogRead(pin);
Function:	read the value from analog pin
Arguments:	pin: A0-A7
Return:	ADC value (0-1023)

Let's design a simple voltage meter using ADC:

```
1  void setup() {
2    Serial.begin(9600);
3    analogReference(DEFAULT);
4  }
5
6  void loop() {
7    // Get the digital value of A0
8    int adc_value = analogRead(A0);
9    // Convert the digital vaule to the real voltage value
10   float v_value = adc_value / 1024.0 * 5.0;
11   Serial.print("The voltage is:");
12   Serial.println(v_value);
13   delay(1000);
14 }
```

4.6 Arduino PWM Wave

PWM wave is used for driving motor, which is mentioned in Chapter 2. The function to setup a PWM wave is:

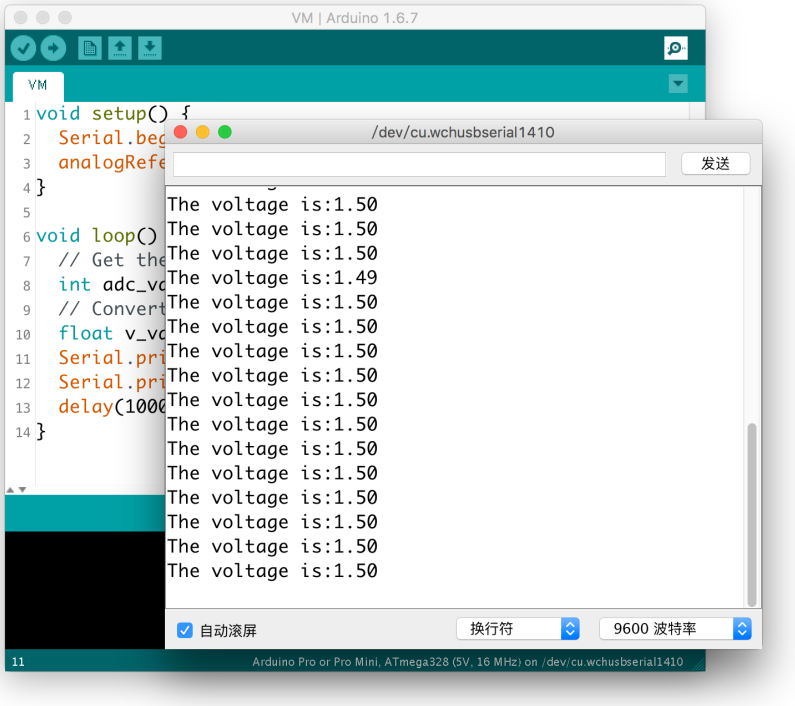


Figure 4.5: The result of voltage meter using ADC

Syntax:	analogWrite(pin, value);
Function:	Writes an analog value (PWM wave) to a pin
Arguments:	pin: 3, 5, 6, 9, 10, and 11 value: the duty cycle: between 0 (always off) and 255 (always on).

After a call to analogWrite(), the pin will generate a steady square wave of the specified duty cycle until the next call to analogWrite() (or a call to digitalRead() or digitalWrite() on the same pin). The frequency of the PWM signal on most pins is approximately 490 Hz. On the Uno and similar boards, pins 5 and 6 have a frequency of approximately 980 Hz. Pins 3 and 11 on the Leonardo also run at 980 Hz. On most Arduino boards (those with the ATmega168 or ATmega328), this function works on pins 3, 5, 6, 9, 10, and 11. You do not need to call pinMode() to set the pin as an output before calling analogWrite().

4.7 Arduino SPI Communication

SPI is another widely used serial communication. It consists of 4 pins, which are SDO (Serial Data Output), SDI (Serial Data Input), SCK (Serial Clock) and CS (Chip Selection).

An example of master sending data

```
1 //Master End
2 #define CK 8
3 #define DO 9
4 #define CS 10
5 void setup() {
```

```
  pinMode(CK,1);    //SCK
  pinMode(DO,1);    //SDO
  pinMode(CS,1);   //CS
  digitalWrite(CK,0);
  digitalWrite(DO,0);
  digitalWrite(CS,1); //0 Selected
}

void loop() {
  // put your main code here, to run repeatedly:
  char dat = 'b';
  unsigned char i = 0;
  digitalWrite(CK,0);
  digitalWrite(CS,0);
  digitalWrite(CK,1);
  for (i = 0; i < 8; i++){
    digitalWrite(CK,0);
    if (dat & 0x80)
      digitalWrite(DO,1);
    else
      digitalWrite(DO,0);
    dat <<= 1;
    delay(1);
    digitalWrite(CK,1);
    delay(1);
  }
  digitalWrite(CS,1);
  delay(1000);
}
```

An example of salver receiving data

```
//Slave End
#define DI 11
#define CS 10
void setup() {
  attachInterrupt(digitalPinToInterrupt(2), get_sck, RISING); //SCK
  // put your setup code here, to run once:
  pinMode(CS, 0);
  pinMode(DI, 0);
  Serial.begin(9600);
}

void get_sck()
{
  static char buf = 0, buf_num = 0, is_cs = 0;
  if (digitalRead(CS) == 0 && is_cs == 0){
    is_cs = 1;
    buf_num = 0;
    buf = 0;
    Serial.println('␣');
  }
  else if(is_cs == 1)
  {
    buf <<= 1;
    buf |= digitalRead(DI)&0x01;
    Serial.print(digitalRead(DI));
```

```
    buf_num++;
    if (buf_num >= 8)
    {
      Serial.print(buf);
      is_cs = 0;
    }
  }
}

void loop() {
}
```

4.8 Arduino Interruption

In Teas ♯1 , the interruption function is used to determine the motor speed. As the motor rotates fast, previous method based on scanning cannot be suitable for getting the pulse generated from IR speed sensor. So the interruption should be used.

Syntax:	attachInterrupt(digitalPinToInterrupt(pin), ISR, mode);
Function:	Setup the interruption
Arguments:	pin: the number of the interrupt 2 and 3 ISR: the ISR to call when the interrupt occurs; this function must take no parameters and return nothing. This function is sometimes referred to as an interrupt service routine. mode: LOW: trigger when pin is low CHANGE: trigger when pin changes value RISING: trigger when the pin goes from low to high FALLING: for when the pin goes from high to low

Let's practice interruption by using an example:

```
int pin = 13;
volatile int state = LOW;

void setup() {
    pinMode(pin, OUTPUT);
    attachInterrupt(digitalPinToInterrupt(2), blink, CHANGE);
}

void loop() {
    digitalWrite(pin, state);
}

void blink() {
    state = !state;
}
```

Chapter 5

Teas ♯1 Testing

It is very difficult to avoid soldering problems. So we need to test each function of Teas ♯1 after finishing soldering. In this chapter, we provide testing codes for some important part, which are LEDs, keys, collision switch, IR sensors, motors and speed sensors. Please load each code to Arduino nano, and test each function. Firstly, please watch the video of testing each function of Teas ♯1 smartcar: Go to Youku.

5.1 Framework of Program

The program Teas ♯1 includes for parts: main program, communication program, motor control program and LED program. The LED program is provided by Adafruit Neopixel Library. Communication program is used to communicate with 74XX165 in order to obtain the sensor information. Motor control program is used to control the motor speed and distance. The user functions should be put in the main program.

The main program in Arduino IDE should be name as xxx.ino, which should be as same as the name of the folder. We open the code of each part and encourage you to program on it.

5.2 Testing of RGB LED

In debugging, the most important part is feedback. The general used feedback ways are LED, LCD and beeper, or communication with MCU using serial port. As the font on a LCD is very small, which cannot be noticed when the car is moving, we select a colorful RGB LED to represent information and state of the smartcar.

WS2812 is a colorful LED which integrated driver, controller. It can be used serially by using only one digital PIN of MCU. In terms of PINs, VCC, GND, DIN and DOUT are on the IC. For MCU, only one PIN should be linked to the first LED's DIN, and the following LED's DIN should be linked to the previous one's DOUT. Finally, all the VCC and GND are connected together and then they can work.

Click to download the code, upzip it, and upload the program to Arduino Nano .

Main program is following:

```
#include "Adafruit_NeoPixel.h"  //Ł
#include "comm.h"               //
#include "motor.h"              //

#define PIN            4
#define NUMPIXELS      2
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

void setup()
{
```

```
    shift_reg_init();   //
    motor_init();       //
    pixels.begin();     //
}
void loop()
{
    pixels.setPixelColor(0, pixels.Color(100,0,0)); //łRGB(0~255)
    pixels.setPixelColor(1, pixels.Color(100,0,0)); //ł
    pixels.show();  //
    delay(1000);    //1
    pixels.setPixelColor(0, pixels.Color(0,100,0));
    pixels.setPixelColor(1, pixels.Color(0,100,0));
    pixels.show();
    delay(1000);
    pixels.setPixelColor(0, pixels.Color(0,0,100));
    pixels.setPixelColor(1, pixels.Color(0,0,100));
    pixels.show();
    delay(1000);
}
```

The testing should be as following:

The color of both left and right LED will change color from red, green to blue. The time interval is 1s. If the LED cannot be lighted, please check the LED is solder well or not or the MCU socket is OK or not.

5.3 Testing of Keys and Collision Switch

There are two free functional keys on Teas ♯1 , which are located beside the battery case. Let's test they can work or not.

Click to download the code, upzip it, and upload the program to Arduino Nano .

Main program is following:

```
#include "Adafruit_NeoPixel.h"  //Ł
#include "comm.h"               //
#include "motor.h"              //

#define PIN            4
#define NUMPIXELS      2
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

void setup()
{
    shift_reg_init();   //
    motor_init();       //
    pixels.begin();     //
}
void loop()
{
    reload_shift_reg(); //
    if(sensor.key_1)    //1//comm.h
    {
        pixels.setPixelColor(0, pixels.Color(100,0,0)); //łRGB(0~255)
        pixels.setPixelColor(1, pixels.Color(100,0,0)); //ł
        pixels.show();  //
    }
```

```
    if(sensor.key_2)    //2
    {
        pixels.setPixelColor(0, pixels.Color(0,150,0));
        pixels.setPixelColor(1, pixels.Color(0,150,0));
        pixels.show();
    }
}
```

The testing should be as following:

Press the left key, the LED will become green and then press the right key, the LED will become red.

There are six collision switch are located surround the PCB board for detecting objects. Let's test them

Click to download the code, upzip it, and upload the program to Arduino Nano .

Main program is following:

```
#include "Adafruit_NeoPixel.h"  //Ł
#include "comm.h"               //
#include "motor.h"              //

#define PIN            4
#define NUMPIXELS      2
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

void setup()
{
    shift_reg_init();   //
    motor_init();       //
    pixels.begin();     //
}
void loop()
{
    char r0,r1,g0,g1,b0,b1;
    reload_shift_reg(); //()
    if(sensor.switcher_back_left) r0 = 100; else r0 = 0;
    if(sensor.switcher_front_left_2) g0 = 100; else g0 = 0;
    if(sensor.switcher_front_left_1) b0 = 100; else b0 = 0;
    if(sensor.switcher_front_right_1) b1 = 100; else b1 = 0;
    if(sensor.switcher_front_right_2) g1 = 100; else g1 = 0;
    if(sensor.switcher_back_right) r1 = 100; else r1 = 0;

    pixels.setPixelColor(0, pixels.Color(r0,g0,b0)); //ŁRGB(0~255)
    pixels.setPixelColor(1, pixels.Color(r1,g1,b1)); //Ł
    pixels.show();  //
}
```

The testing should be as following:

Press all the switch from left to right and from up to down. The near LED will show green, blue, blue, red and red. If the LED is not on, check the pull-up resistor of the switch is solder or not and check the direction of switch. If all the switch cannot work, please check the 74HC165 IC is inserted or not and the direction of it.

5.4 Testing of IR Tracking Sensor

IR sensor is used to detect the darkness of the ground. In Teas ♯1 , seven IR sensors are used. Let's start to test them.

Click to download the code, upzip it, and upload the program to Arduino Nano .

Main program is following:

```
#include "Adafruit_NeoPixel.h"  //Ł
#include "comm.h"               //
#include "motor.h"              //

#define PIN            4
#define NUMPIXELS      2
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

void setup()
{
    shift_reg_init();   //
    motor_init();       //
    pixels.begin();     //
}
void loop()
{
    char r0,r1,g0,g1,b0,b1;
    reload_shift_reg(); //
    if(sensor.ir_left_3) //ł()
        r0 = 100; //ł1
    else
        r0 = 0;
    if(sensor.ir_left_2) g0 = 100; else g0 = 0;
    if(sensor.ir_left_1) b0 = 100; else b0 = 0;
    if(sensor.ir_right_1) b1 = 100; else b1 = 0;
    if(sensor.ir_right_2) g1 = 100; else g1 = 0;
    if(sensor.ir_right_3) r1 = 100; else r1 = 0;
    if(sensor.ir_mid) r0=r1=g0=g1=b0=b1 = 100;

    pixels.setPixelColor(0, pixels.Color(r0,g0,b0)); //łRGB(0~255)
    pixels.setPixelColor(1, pixels.Color(r1,g1,b1)); //ł
    pixels.show();  //
}
```

The testing should be as following:

Put the smartcar on light color desk and stick one black tape (around 2.5cm width). Let each sensor of the smartcar over the tape, if sensors can work, the color of LED will be red, yellow, green, blue, white, blue, yellow and red. You can tell which one cannot work with the color.

For sensors that can detect white but can not detect black, use a black whiteboard to slightly black the sensor's tube, because the sensor's current limiting resistor is too small and the infrared light is too strong, causing the black surface to return Sufficient infrared light, the receiver is considered light. If you can detect black, but can not detect the white, indicating that the infrared light is too weak, may be too long welding lead to damage to the infrared diode, you can consider replacing the sensor. The normal operation of the infrared sensor is very important for subsequent tracing functions, so be sure to put the trolley on the ground for testing.

5.5 Testing of Motor

Click to download the code, upzip it, and upload the program to Arduino Nano .

Main program is following:

```
#include "Adafruit_NeoPixel.h"  //Ł
```

```
#include "comm.h"                //
#include "motor.h"               //

#define PIN            4
#define NUMPIXELS      2
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

void setup()
{
    shift_reg_init();    //
    motor_init();        //
    pixels.begin();      //
}
void loop()
{
    static int motor_right;
    static int motor_left;
    reload_shift_reg(); //()
    if(sensor.key_1)
    {
        while(sensor.key_1) reload_shift_reg(); //Ł
        delay(20);
        if(motor_right == 0)
            motor_right = 150;
        else if(motor_right == 150)
            motor_right = -150;
        else if(motor_right == -150)
            motor_right = 0;
    }
    if(sensor.key_2)
    {
        while(sensor.key_2) reload_shift_reg(); //Ł
        delay(20);
        if(motor_left == 0)
            motor_left = 150;
        else if(motor_left == 150)
            motor_left = -150;
        else if(motor_left == -150)
            motor_left = 0;
    }
    motor_set_PWM(motor_left,motor_right); //(,)(0~255)
}
```

The testing should be as following:

Press the left button once, the left side of the motor is running, press again, the motor reverses, press again, the motor stops. Right to control the right side of the motor. Forward rotation is defined as a way to move the car forward. If you find the direction of reversing the positive and negative, please change the direction of the motor line, or modify the program to achieve the normal direction of rotation of the motor.

5.6 Testing of Speed Sensor

Click to download the code, upzip it, and upload the program to Arduino Nano .

Main program is following:

```
#include "Adafruit_NeoPixel.h"  //Ł
#include "comm.h"               //
#include "motor.h"              //

#define PIN            4
#define NUMPIXELS      2
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

void setup()
{
    shift_reg_init();   //
    motor_init();       //
    pixels.begin();     //

    pixels.setPixelColor(0, pixels.Color(0,100,0));
    pixels.setPixelColor(1, pixels.Color(0,100,0));
    pixels.show();
}
void loop()
{
    static int state = 1;
    static int state_now = 0;
    reload_shift_reg(); //()

    if(left_pulse%2)    //Ł
        pixels.setPixelColor(0, pixels.Color(0,0,100));
    else
        pixels.setPixelColor(0, pixels.Color(0,100,0));
    if(right_pulse%2)
        pixels.setPixelColor(1, pixels.Color(0,0,100));
    else
        pixels.setPixelColor(1, pixels.Color(0,100,0));
    pixels.show();

    if(sensor.switcher_back_right)  //360
    {
        motor_step(0,150,0,48);//ł48
        motor_step(0,0);//
    }

    if(sensor.switcher_back_left)   //360
    {
        motor_step(150,0,48,0);//ł48
        motor_step(0,0);//
    }
}
```

The testing should be as following:

First turn the wheel to see if the corresponding position of the LED is flashing, if not blinking, adjust the blue and white adjustable potentiometer, and continue to turn the wheels until both sides of the LED are flashing. Place the car on the floor, press the left button once, turn the left wheel to rotate the car for a week, right control the right side of the motor, so that the car rotation for a week. Finished test.

If you find the car has been ring, can not stop, indicating that the sensor may not be normal welding, please check the solder joints; you can also use the oscilloscope to observe the sensor output waveform to determine the problem. If you find that there is no turn to stop a week, it may be sensor signal is not normal, this time need to continue to adjust the potentiometer, it is best to use the oscilloscope to observe

the waveform to determine when the car is not moving, the sensor is not output signal.

5.7 Exercise: Write a complete program to test all the features

The above sections, we have been tested the various parts of the car, the following readers through the above examples, their own preparation of a complete program, test Teas ♯1 all parts of the function. You can consider using the left and right keys to select the part of the test, for example, left after the end of the self-test, it is used to switch to the next test part, right to switch the same test part of the different sensors. Ask the reader to think about how to combine the above five programs together.

Chapter 6

Teas ♯1 Application

6.1 Black Line Tracking

Automatic tracking is one of the useful application for Teas ♯1 . The smartcar can track the 2.5cm width black line and move forward. In order to achieve this application, IR sensors should be used to determine the color of ground. There are many methods to track the black line. A simple way is that turn the car to the direction that the black line is located. By using this simple method, you can develop a smartcar like the video shown: `http://v.youku.com/v_show/id_XMTQ3Nzg4Mjk0OA`.

6.2 Impact Avoidance and Falling Prevention

The IR sensors and impact switchers are used to finish this application. A simple method is that when the IR sensors determine the color is black or the switcher determine impact, move backward a litter. Then turn to the direction of such sensor.

By using this simple method, you can develop a smartcar like the video shown: `http://v.youku.com/v_show/id_XMTQ3NzkwNTg2MA`.

6.3 More Ideas

Let's open the mind and think about more new ideas based on Teas ♯1 .

Appendix A

Schematics and PCB

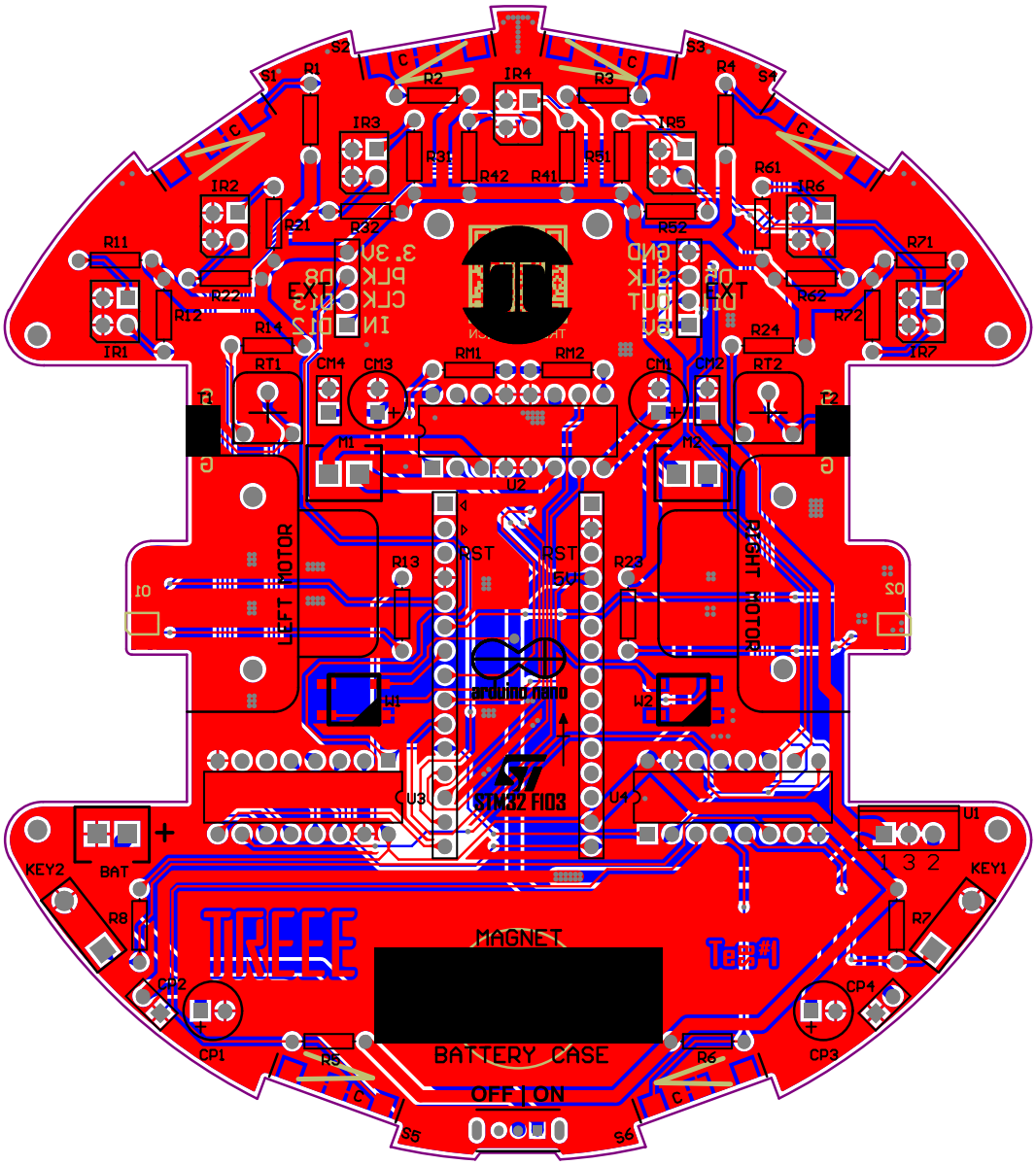


Figure A.1: Teas ♯1 PCB

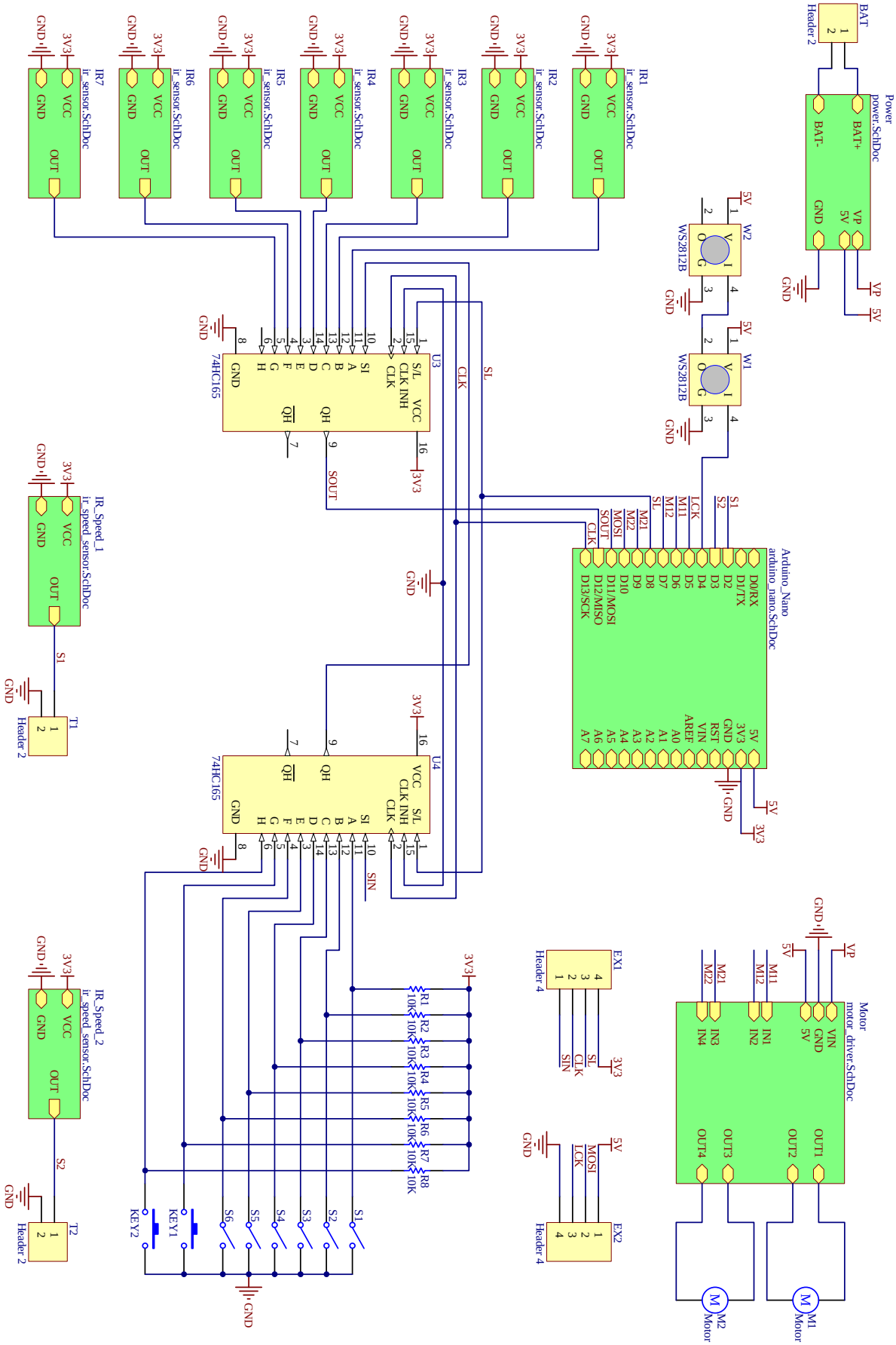


Figure A.2: Teas ♯1 Schematics-Main

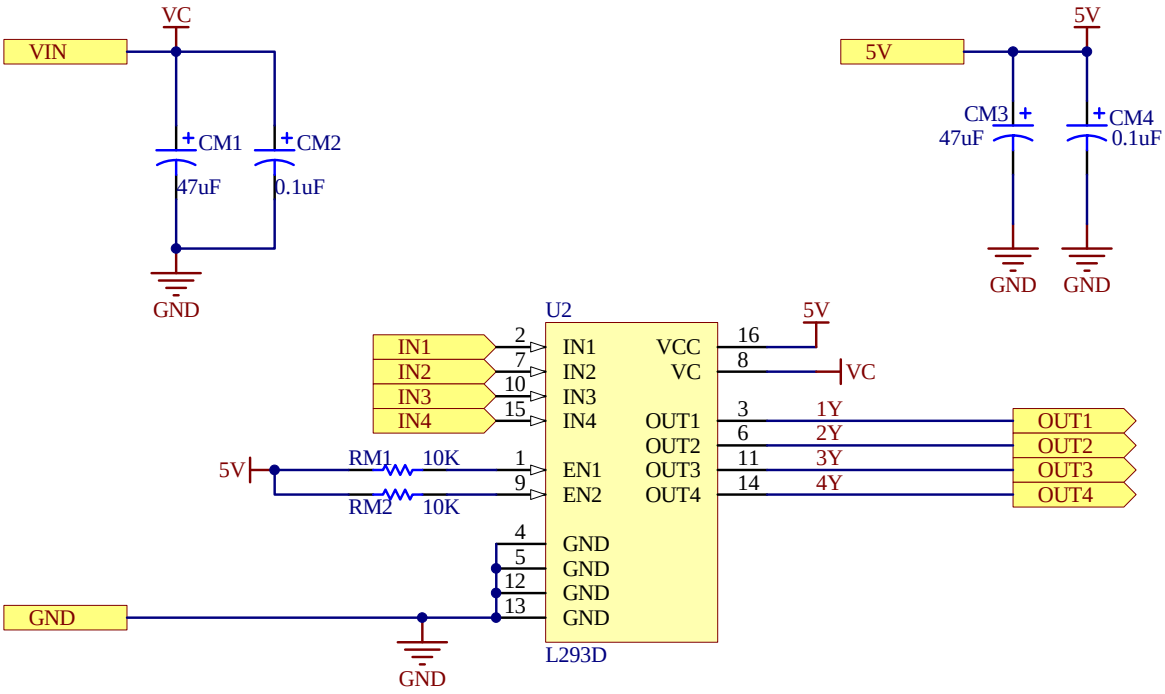


Figure A.3: Teas ♯1 Schematics-Motor Driver

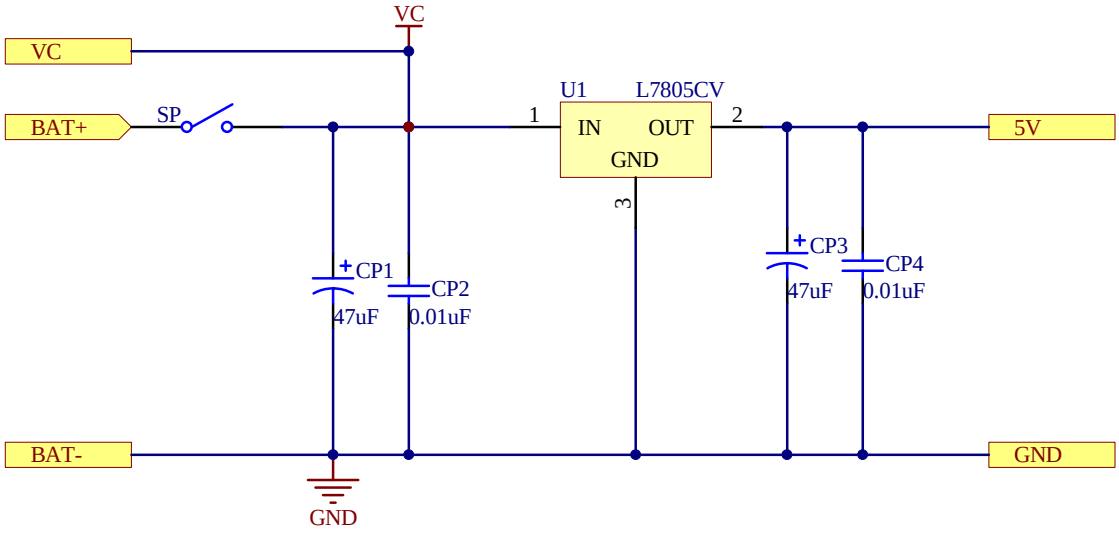


Figure A.4: Teas ♯1 Schematics-Power

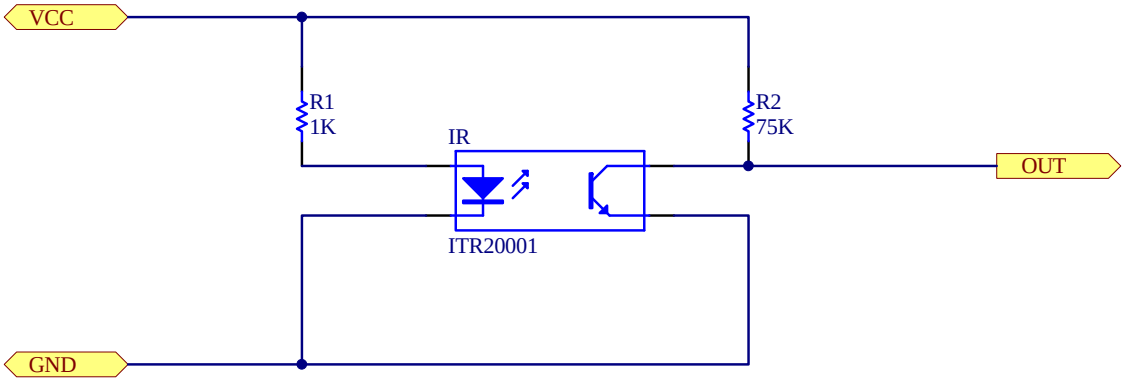


Figure A.5: Teas ♯1 Schematics-IR Sensor

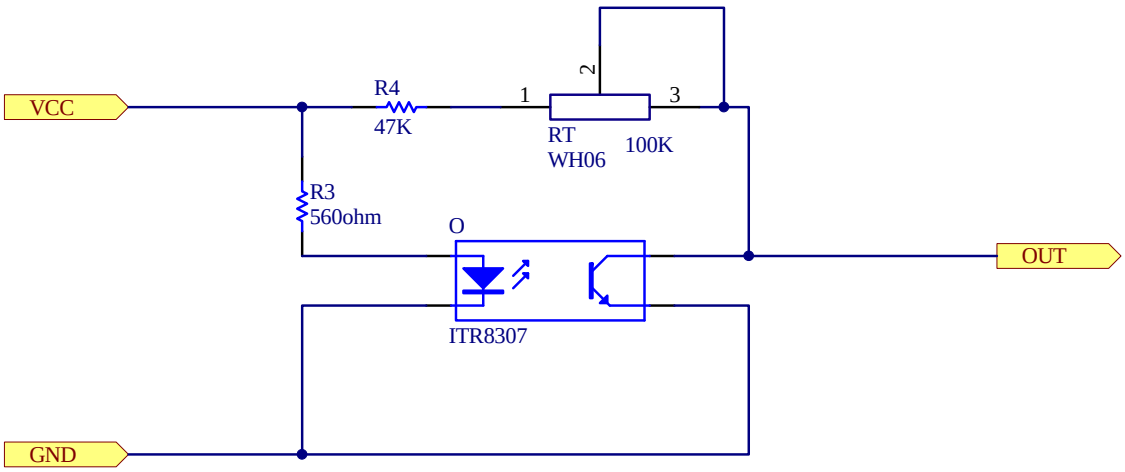


Figure A.6: Teas ♯1 Schematics-IR Speed Sensor

Appendix B

Disclaimers