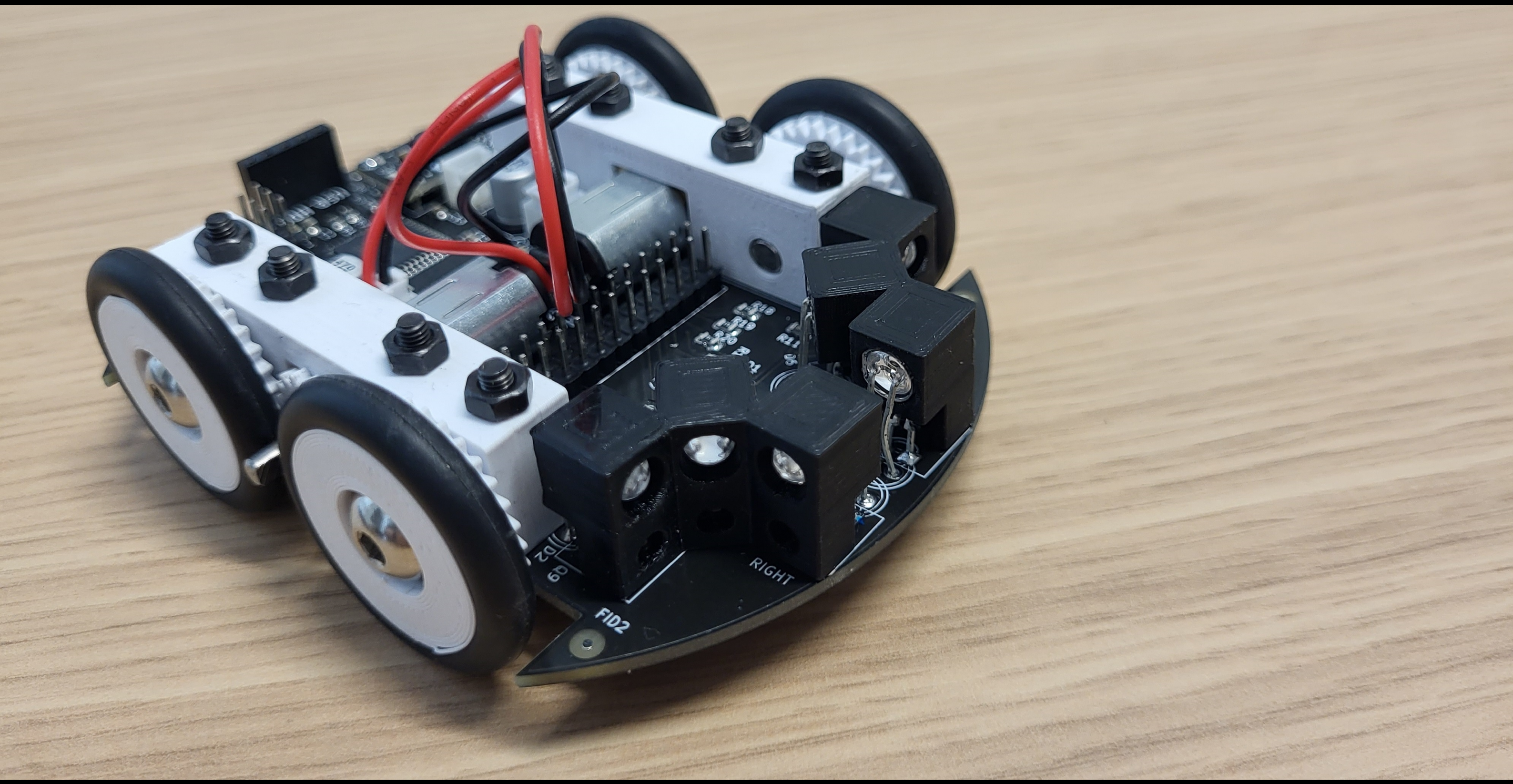
Simple Mouse Guide



Ken Cameron

Contents

Introduction

SimpleMouse is a small, self-contained robotic device intended to be used in standard size Micromouse mazes.

- It uses the ATmega32U4 processor running at 16Mhz. This provides a USB interface for programming.
- It makes use of 6V N20 motors with attached gearboxes. These are available in a range of RPM. They can be driven via PWM (Pulse Width Modulation) to vary speed.
- It uses hall effect sensors to detect wheel rotation.
- It uses infra-red sensors to detect proximity of objects such as maze walls.
- An multi-colour addressable LED is provided to show status.
- It can use batteries that can supply voltage in the range 3-4.5v. Lithium-Polymer 3.7v or three AA cells providing 4.5v are recommended.
- It has a bootloader and runtime environment compatible with the Arduino IDE.
- It has an expansion header allowing additional hardware to be connect via a daughter board.

1.1 Processor

The SimpleMouse uses the ATmega32u4 micro-controller from Microchip. This provides 32KBytes of flash memory for code, 2.5KBytes of ram for data and 1KByte of EEPROM for storage of persitent data.

A bootloader is provided that allows program download over the integrated USB connection. This reduces the available flash memory for user programs to around 28 KBytes.

1.2 Drivetrain

1.3 Wall Sensors

1.4 Status LED

A single addressable LED is provided to allow status to be indicated. The package is drived by a single wire using a serial encoding that allows the brightness of the red, green and blue leds it contains to be set independently.

Development Environment

The SimpleMouse uses the same processor as the Arduino Leonardo and has been designed to be able to use the same development environment. However, not all pins serve the same functions on both platforms so software compiled for the Leonardo should not be used on the SimpleMouse.

The key difference is that the SimpleMouse does not provide the TX,RX and LED leds of the Leonardo. Instead it provides a single addressable LED. Some of the pins used to drive the leds on the Leonardo are inputs on the SimpleMouse and should not be driven.

2.1 Bootloader

The Arduino IDE makes use of the Caterina bootloader to download sketches to ATmega32U4 based boards. A modifed version of this bootloader is provided for SimpleMouse. The current version does not provide status via the leds.

The custom bootloader needs to be downloaded using the ISP header with a suitable programming device. It cannot be done via USB. This should only need to be done once after the mouse pcb is assembled. This requires an external programming device connected to a host computer. The hardware fuses should also be set at this time.

2.2 IDE

A custom firmware is available for the SimpleMouse that is a modified version of the Arduino Leonardo firmware. This allows the Arduino IDE to be used to build sketches and download them to the SimpleMouse.

The software package required should be installed in the hardware/ directory of the users local Arduino directory. This is the same place where libraries the user has installed locally are kept. When this is done, Bath Simple Mouse should appear in the list of target platforms available.

Drive System

3.1 Mechanical

The SimpleMouse contains two independent drive trains. Each uses a 6v N20 motor with a fixed gearbox attached. These motor plus gearbox units are commodity items that come in a range of gearbox ratios in a range from about 15rpm to 3000rpm. You are advised to start with a slower speed when doing initial development and move to higher speed variants when your software is further developed. Starting in the range 500rpm or lower is recommended.

The motors drive a small cog that drive two wheels in the same direction. The two drivetrains are located on either side of the SimpleMouse and position so that when driven in opposite directions the SimpleMousewill rotate about its center.

3.2 Pulse-Width Modulation

SimpleMouse uses the DRV8833 motor driver chip. This is a two channel H-Bridge device that allows the two motors to be fully independent and driven in either direction. Each motor is driven by two control signals. When the level of each signal is the same, the motors do not turn. When they are different the motor turns. Which way the motor turns depends on which of the two signals is high. The controller operates with different levels of decay when both pins are set the same. When they are both zero, the decay is fast, referred to by the manufacturer as 'coast'. Then they are both one the decay is slow, referred to as 'brake'.

When the pins are set to different values to drive the motor, the powered pin can be driven with a PWM signal. This will vary the speed of the motor. All four pins selected to drive the pair of motors have hardware support for PWM signal generation. In a sketch, the analogWrite() function can be used to drive them.

3.3 Rotation Detection

One wheel on each drivetrain contains 16 small embedded magnets. A Hall Effect sensor is available is able to detect the passing of each magnet to provide wheel rotation feedback.

Infra-Red Sensors

4.1 Emitters

4.2 Photo-Diode Sensing

Expansion Headers

The SimpleMouseprovides three headers.

5.1 USB

1	2	3	4	5
+5V	D-	D+	nc	GND

Figure 5.1: USB Header

This connector is a standard 0.1in SIL header. It will require an adapter to provide a USB B socket.

5.1.1 Power

The USB power GND and +5v is assigned to pins 5 and 1 respectively. Note that this power is only supplied to the USB core. It does not supply the reset of the device. The processor core require a 5v supply either via the battery or pin 11 or 12 of the expansion header from a fly lead on the USB adapter.

If supplied via a battery, the motor will also be functional. If supplied via the expansion header it will not.

You should not connect 5v to the expansion header while the battery is connected. This can damage the SimpleMouse.

5.1.2 Data

The USB data connections -D and + are assigned to pins 2 and 3 respectively. Pin 4 is not connected. The ID capability is not supported.

5.2 ISP

MISO	1	2	+5V
SCLK	3	4	MOSI
RESET	5	6	GND

Figure 5.2: AVR ISP Header

The ISP header is a standard 6 pin DIL 0.1in header. It follows the AVR defined pinout for in-system programming of the ATmega32u4. When used for programming, no battery or USB host should be connected.

5.3 Expansion

ENCB	BIN2	A4	EM1	EM3	+5V	GND	MSLEEP	TXD1	SDA	PB4	A1	AIN2
2	4	6	8	10	12	14	16	18	20	22	24	26
1	3	5	7	9	11	13	15	17	19	21	23	25
ENCA	BIN1	A5	A3	EM2	+5v	GND	HWB	RXD1	SCL	A2	A0	AIN1

Figure 5.3: Expansion Header

The expansion header brings out all of the micro-controller I/O pins used by the SimpleMouse as well as a power supply. In addition, it also provides connection to the I2C bus and second serial channel of the microcontroller.

It is intended to be used in one of three ways.

1. (Normal) The ATmega32U4 is in full control and most pins can be monitored but should not be driven. However, the TXD1, RXD1, SCL, SDA, PB4 and HWB pins can be used by the ATmega32U4 to control additional circuitry.
2. (Slave) With a suitable sketch loaded, the ATmega32U4 can accept commands over the Serial connection (TXD1, RXD1) to act as a slave controller to a host processor.
3. (Chassis) With the ATmega32U4 not present or all I/O configured as input, it provides a chassis controllable by another processor. A second processor can control the motors and read the inputs directly. When no ATmega32U4 is present, the link LEDHWB can be closed on the pcb and the HWB pin will become the LED input.

11,12	+5V	+5V output when powered by battery. A +5v input when powered by USB. Whne used as an input, power is not supplied to the motor. Do not connect power to these pins when the battery is also connected, it may damage the SimpleMouse.
13,14	GND	Ground connection.
1	ENCA	The left Hall Effect sensor.
2	ENCB	The left Hall Effect sensor.
23,24,21,7,6,5	A0,A1,A2,A3,A4,A5	The analog inputs of the wall sensor photo-diodes.
25,26	AIN1,AIN2	The motor control pins for the right motor.
3,4	BIN1,BIN2	The motor control pins for the left motor.
16 19,20	MSLEEP SCL, SDA	This signal turns the motor controller chip on or off. 0 = off, 1 = on. The I2C bus of the ATmega32U4
17,18	RXD1, TXD1	The second serial port of the ATmega32U4. Access via Serial1 in software.
8,9,10	EM1,EM2,EM3	Control signals for the IR emitter pairs. 0 = off, 1 = 0n. EM1 = FORWARD, EM2 = SIDE, EM3 = ANGLE.
22	PCB	Available I/O pin, not assigned any function.
33	HWB	Hardware Boot PIN. When used with AU variant of the chip, provides access to the factory bootloader when held low on reset. Otherwise, can be used as a general I/O pin. When no processor is present on pcb, and the LEDHWB link is made, this pin maps to the LED input.

Figure 5.4: Expansion Functions

A. Layout and Schematic

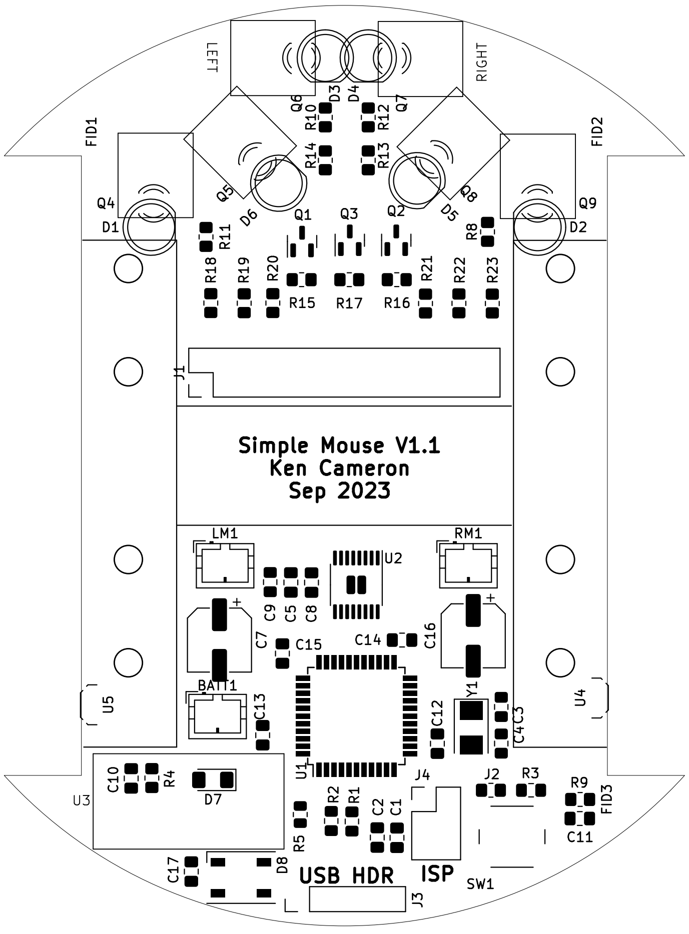


Figure 5: SimpleMouse Layout

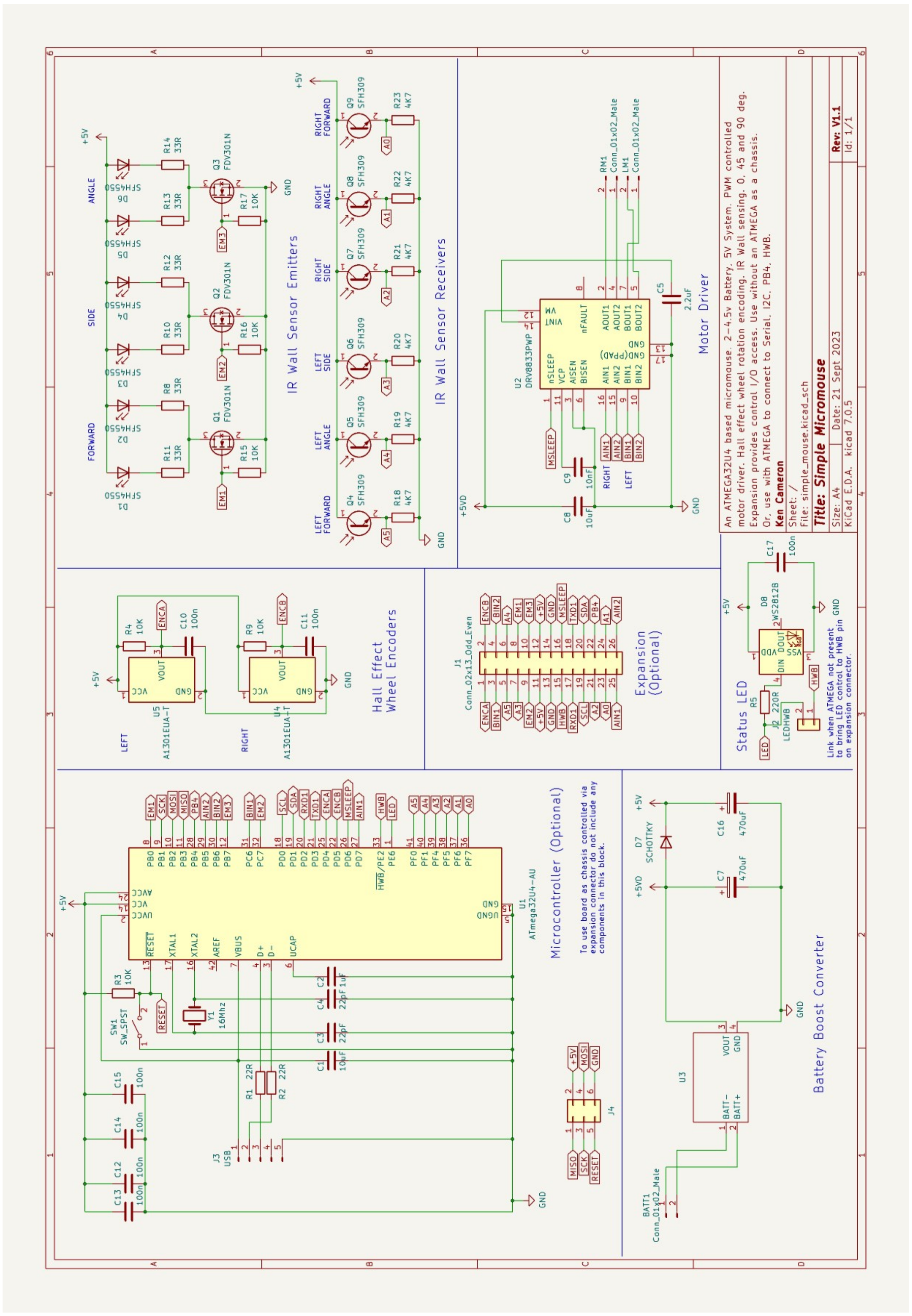

IR Wall Sensor Emitters
IR Wall Sensor Receivers
Motor Driver
Hall Effect Wheel Encoders
Expansion (Optional)
Status LED
Microcontroller (Optional)
To use board as chassis controlled via expansion connector do not include any components in this block.
Battery Boost Converter
Link when ATMEGA not present to bring LED control to HWB pin on expansion connector.
An ATMEGA32U4 based micromouse, 2-4.5v Battery, 5V System, PWM controlled motor driver, Hall effect wheel rotation encoding, IR Wall sensing, 0, 45 and 90 deg.
Expansion provides control I/O access. Use without an ATMEGA as a chassis.
Or, use with ATMEGA to connect to Serial, I2C, PB4, HWB.
Ken Cameron
Sheet: /
File: simple_mouse.kicad_sch
Title: Simple Micromouse
Size: A4
Date: 21 Sept 2023
Rev: V1.1
KiCad E.D.A. kicad 7.0.5
Id: 1/1

Figure 6: SimpleMouse Schematic

B. Pin Mapping

Function	Port	C Name	Dir.
AIN1	PD7	D6	Out
AIN2	PB5	D9	Out
BIN1	PC6	D5	Out
BIN2	PB6	D10	Out
MSLEEP	PD6	D12	Out
EM1	PB0	D7	Out
EM2	PC7	D13	Out
EM3	PB7	D11	Out
A0	PF7	A0	In
A1	PF6	A1	In
A2	PF5	A2	In
A3	PF4	A3	In
A4	PF1	A4	In
A5	PF0	A5	In
ENCA	PD4	D4	In
ENCB	PD5	D30	In
LED	PE6	D7	Out
TXD1	PD3	D1	Out
RXD1	PD2	D0	In
SCL	PD0	D15	Out
SDA	PD1	D2	In/Out
PB4	PB4	D8	In/Out
HWB	PE2	–	In/Out

Figure 7: Pin mapping

C. Known Issues

.1 Hall Encoders have no interrupt support.

.1.1 Issue

The pins used have no change interrupt support. The state can only be polled.

.1.2 Suggested Workaround

Cut tracks and add jumper wires to match recommended fix.

Or, Use an additional xor gate to connect ENCA and ENCB to PB4.

.1.3 Recommended Fix

Adjust design to map ENCA and ENCB to PB0 and PB4. Map EM1 to PD4. PB4 replaced on expansion header with PD5.