I/O mapping between Arduino Nano and Raspberry Pi Pico Design

Rob Probin, Dec 2022

Based on David Hannaford's Pico build:
https://ukmars.org/2021/02/a-raspberry-pi-pico-based-ukmarsbot/

This design has some differences; apart from the battery voltage divider ratio (22K:10K approx 1/3 rather than 10K:10K = 1/2), and the requirement to use a 74HC86 instead of a 74LS86, there are no other modifications required to the base board. See http://zedcode.blogspot.com/2022/12/converting-ukmarsbot-to-raspberry-pi.html fro some details.

The sensor board ***may*** require adjustment in IR LED output or phototransistors.

You can swap the module with an Arduino Nano and the design will still work.

GPIO Numbering of the Pico has been maintained where possible to map to the Nano, so two set of numbers don't need to be memorised. See D2 on the Arduino maps to GP2 on the Pico. Another instance is D13/GP13 (and this is a specific difference from DH Design).

However, TX/RX are reversed, so this has been swapped on the first two pins, like DH design.

Nano Pinout connections

Name and GPIO no	Role	Nano pin	Role	Same mapping as DH	Pico pin no	Pico GP no
int Transmit = D1:	// Transmit pin	1	Out	Yes		GP0/TX
int Receive = D0;	// Receive pin	2	In	Yes		GP1/RX
Reset		3	In	Yes		n/c
Gnd		4		Yes		GND
int m1encoder1 = D2;	// motor 1 encoder 1 input interrupt pin	5	In	Yes		GP2
int m2encoder1 =D3;	// motor 2 encoder 1 input interrupt pin	6	In	Yes		GP3
int m1encoder2 = D4;	// motor 1 encoder 2 input	7	In	Yes		GP4
int m2encoder2 = D5;	// motor 2 encoder 2 input	8	In	Yes		GP5
int sensorLED1 = D6;	// 1st diagnostic LED on sensor board	9	Out	Yes		GP6
int lmotorDIR	// Left motor dirn	10	Out	Yes	10	GP7

= D7;	input 1					
int rmotorDIR = D8;	// Right motor dirn input 3	11	Out	Yes	11	GP8
int lmotorPWM = D9;	// Left motor PWM pin	12	Out	Yes	12	GP9
int rmotorPWM = D10;	// Right motor PWM pin	13	Out	Yes	14	GP10
int sensorLED2 = D11;	// 2nd diagnostic LED on sensor board	14	Out	Yes		GP11
int trigger = D12;	// trigger for sensor LEDs	15	Out	Yes		GP12
int LED13 = D13;	// ext LED Red (Buzzer)	16	Out	No		~~GP20~~ GP13
+3.3V	Regulated down from Vin	17	Out	Yes		
Aref		18	In	Yes		
int lside = A0;	// left side sensor input	19	In	No		4051
int rfront = A1;	//front left line sensor input	20	In	No		4051
int lfront = A2;	//front right left sensor input	21	In	No		4051
int rside = A3;	// right side sensor input	22	In	No		4051
int sens1 = A4;	// unassigned sensor input 1	23	In	No		4051
int sens2 = A5;	// unassigned sensor input 2	24	In	No		4051
int fourwayswitch = A6;	// input from function switch	25	In	No		4051
int battery = A7;	// input for battery measurement	26	In	No		4051
+5V	Regulated down from Vin	27	Out	?		3.3v out
RESET		28	In	Yes		n/c
GND		29		Yes		GND
Vin 7 to 12V	Battery IN	30	In	No???		to 5v reg

Other Raspberry Pi Pico Connections

These do not map to DH's design.

Name and GPIO	**Role**		
GP20 Analogue Select Output	HC4051 Select S0		

GP21 Analogue Select Output	HC4051 Select S1		
GP22 Analogue Select Output	HC4051 Select S2		
GP23 ADC2	Analogue input		
GP14 I2C1_SDA	IMU SDA		
GP15 I2C1_SCL	IMU SCL		
GP16-GP19	Optional SDCard SPI interface		

Other 5v requirements:

- Bluetooth Serial Power HC05/HC06 requires 5v power, but has 3.3v TX/RX signals
- IMU power – often these have on board regulators that require 5v, although the signals (SPI or I2C) will be 3.3v. I am using a MPU-9250 device for testing.
- Raspberry Pi Pico Vsys – we can run this from 1.8v to 5.5v because of the buck-boost on the Raspberry Pi Pico. However, it cannot take the 8-9v of the PP3 battery … so we require a regulator. We use the same feed as the other 5v devices for this.
  - NOTE: We fit a schottky diode as suggested in the Raspberry Pi Pico manual.