Arduino

Basic Software

Items to cover

- Programming methods
  - Integrated Development Environment (IDE)
  - Flow chart (XOD)
- Basic code generation.
- Review of the Arduino IDE
- Example program to control a stepper motor

Programming Methods

- There are a number of ways to program an Arduino.
  - Machine code
    - Old school.
      - Very efficient in memory usage.
      - **Very** difficult.
  - Use an Integrated Development Environment (IDE).
    - Most common method.
      - Reasonably efficient in memory usage.
      - Reasonably easy to use.
  - Flow chart
    - Newly available system.
      - Less efficient in memory usage.
      - Easy to understand.

Integrated Development Environment (IDE)

- The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.
- Arduino IDE can run on most software platforms, Windows, Linux, Mac etc.
- Current IDE for the Arduino is;
  - Arduino IDE 1.8.16 and can be downloaded from;
    - Software | Arduino

Integrated Development Environment (IDE)

- Using the IDE software it is possible to prepare the code as a series of ‘English’ statements.
- The IDE is a text editor with a verification function and a compiler to convert the text into a set of instructions.
- The text statements are in the form of C and C++ language but also include unique statements to Arduino.
  - Additional functions are made available by loading ‘Libraries’.
- These statements are then compiled into machine code and uploaded to the Arduino board.
- Syntax is critical when using the IDE system, as I will explain later.

IDE Standard Functions

Digital I/O

digitalRead()
digitalWrite()
pinMode()

Analog I/O

analogRead()
analogReference()
analogWrite()

Zero, Due & MKR Family

analogReadResolution()
analogWriteResolution()

Math

abs()
constrain()
map()
max()
min()
pow()
sq()
sqrt()

Trigonometry

cos()
sin()
tan()

Random Numbers

random()
randomSeed()

Bits and Bytes

bit()
bitClear()
bitRead()
bitSet()
bitWrite()
highByte()
lowByte()

External Interrupts

IDE Standard Functions

Advanced I/O

noTone()
pulseIn()
pulseInLong()
shiftIn()
shiftOut()
tone()

Time

delay()
delayMicroseconds()
micros()
millis()

Characters

isAlpha()
isAlphaNumeric()
isAscii()
isControl()
isDigit()
isGraph()
isHexadecimalDigit()
isLowerCase()
isPrintable()
isPunct()
isSpace()
isUpperCase()
isWhitespace()

External Interrupts

attachInterrupt()
detachInterrupt()

Interrupts

interrupts()
noInterrupts()

Communication

Serial
Stream

USB

Keyboard
Mouse

IDE Standard Functions

Constants

HIGH | LOW
INPUT | OUTPUT | INPUT_PULLUP
LED_BUILTIN
true | false
Floating Point Constants
Integer Constants

Conversion

(unsigned int)
(unsigned long)
byte()
char()
float()
int()
long()
word()

Data Types

array
bool
boolean
byte
char
double
float
int
long
short
size_t
string
String()
unsigned char
unsigned int
unsigned long
void
word

Variable Scope & Qualifiers

const
scope
static
volatile

Utilities

PROGMEM
sizeof()

IDE Standard Functions

Sketch

loop()
setup()

Control Structure

break
continue
do...while
else
for
goto
if
return
switch...case
while

Further Syntax

#define (define)
#include (include)
/* */ (block comment)
// (single line comment)
; (semicolon)
{} (curly braces)

Arithmetic Operators

% (remainder)
* (multiplication)
+ (addition)
- (subtraction)
/ (division)
= (assignment operator)

Comparison Operators

!= (not equal to)
< (less than)
<= (less than or equal to)
== (equal to)
> (greater than)
>= (greater than or equal to)

Boolean Operators

! (logical not)
&& (logical and)
|| (logical or)

Pointer Access Operators

& (reference operator)
* (dereference operator)

Bitwise Operators

& (bitwise and)
<< (bitshift left)
>> (bitshift right)
^ (bitwise xor)
| (bitwise or)
~ (bitwise not)

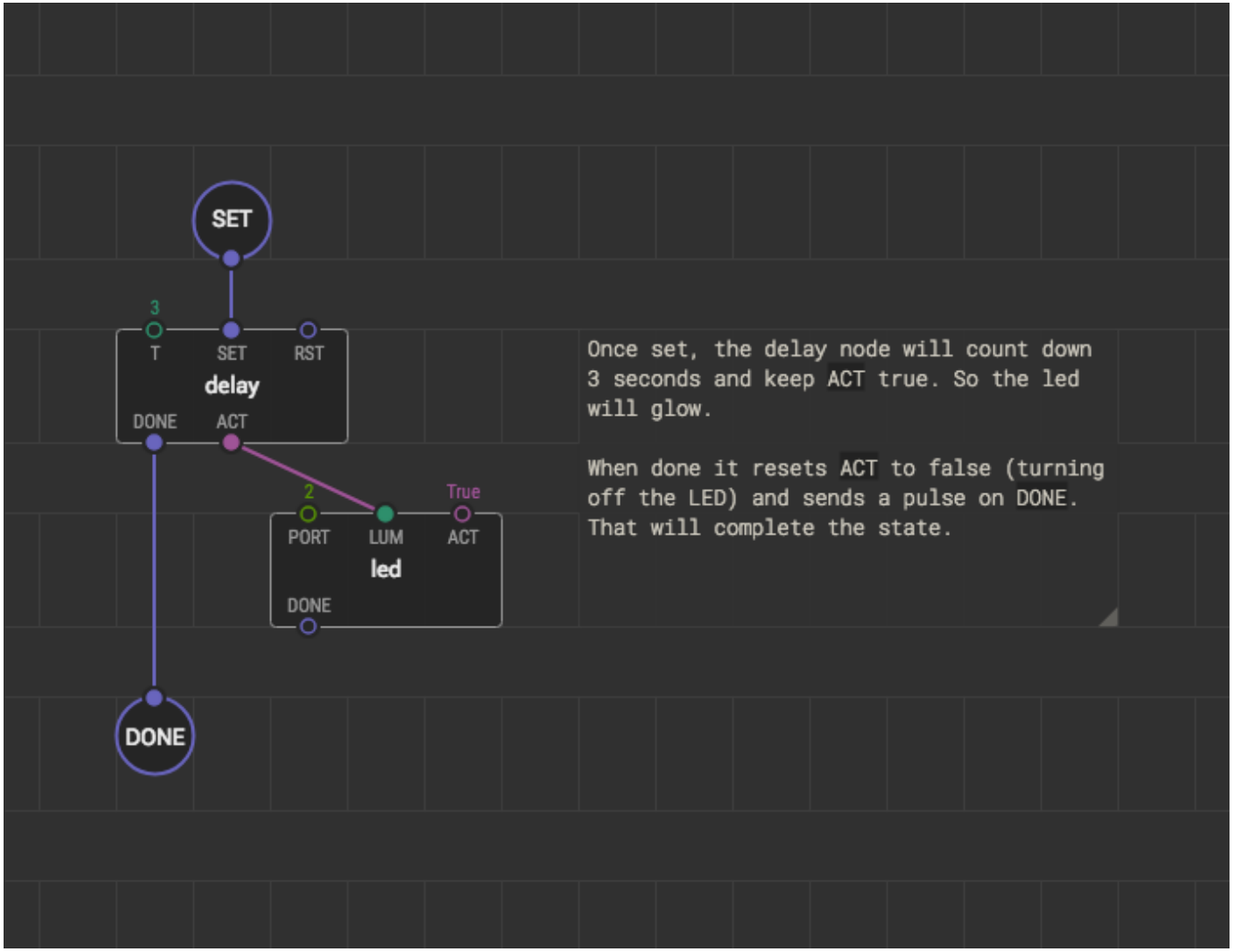
Compound Operators

%= (compound remainder)
&= (compound bitwise and)
*= (compound multiplication)
++ (increment)
+= (compound addition)
-- (decrement)
-= (compound subtraction)
/= (compound division)
^= (compound bitwise xor)
|= (compound bitwise or)

Flowchart IDE

- Flowchart IDE is a graphical method to generate program code.
- It uses graphical building block symbols which are interlinked as required.
- These interlinked blocks are then converted by the program to C++ language, compiled and uploaded to the Arduino board.
- An example of a free flowchart system is XOD and can be downloaded from
  - XOD

XOD example


SET
3
T
SET
RST
delay
DONE
ACT
2
True
PORT
LUM
ACT
led
DONE
DONE
Once set, the delay node will count down 3 seconds and keep ACT true. So the led will glow.
When done it resets ACT to false (turning off the LED) and sends a pulse on DONE. That will complete the state.

Basic program code generation

- In generating software code it is good practice to use a paper flow chart method to develop the requirements irrespective of the manner in which the final code will be developed.
- This enables the flow of commands and instructions to be developed.
- I will develop a simple demonstration program to control a stepper motor to show the use of paper flow charts.

IDE program segments

- The IDE software is divided into 3 distinct segments;
  - Initialization
  - Setup
  - Loop
- These segments have defined functions when developing a programs.

IDE Initialisation

- The initialisation section of the program must be at the beginning of the code and is only run once.
- In this section we can typically do the following;
  - Load Libraries
  - Set initial values for variable.
  - Configure Library functions

IDE Setup

- The Setup section of the program must follow the Initialisation section of the code and is only run once.
- In this section we can typically do the following;
  - Define Inputs and Outputs (IO)
  - Start serial communications
  - Load variable into library functions

IDE Loop

- This is the main section of the program and must follow the initialisation and setup sections, it runs constantly as a loop.
- It can be a single loop or a series of nested loops.
- It will continue to run until either;
  - The power is removed
  - The reset button is pressed
    - If the reset button is pressed the program will start with the initialisation section.

Example program

- Develop a simple program to control the position of a stepper motor to follow the position of a potentiometer, this could be a simple antenna rotator.
- We will use a Library called Stepper.h. This is a set of additional functions specific to simple stepper motors and provides the onal functions.

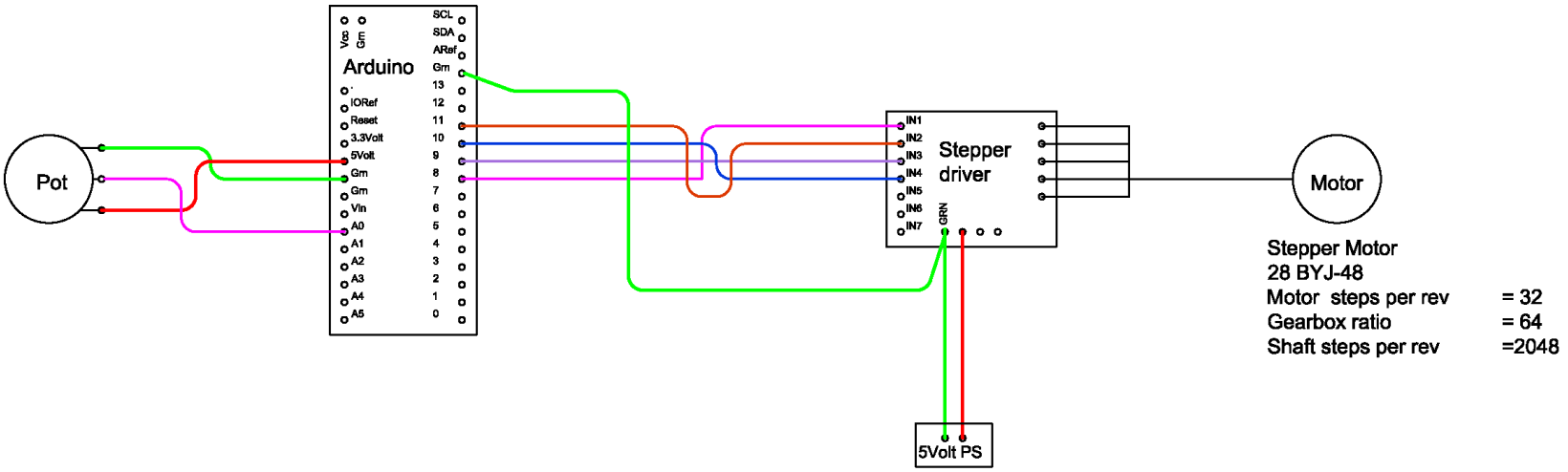
Functions

- Stepper(steps, pin1, pin2)
- Stepper(steps, pin1, pin2, pin3, pin4)
- setSpeed(rpm)
- step(steps)

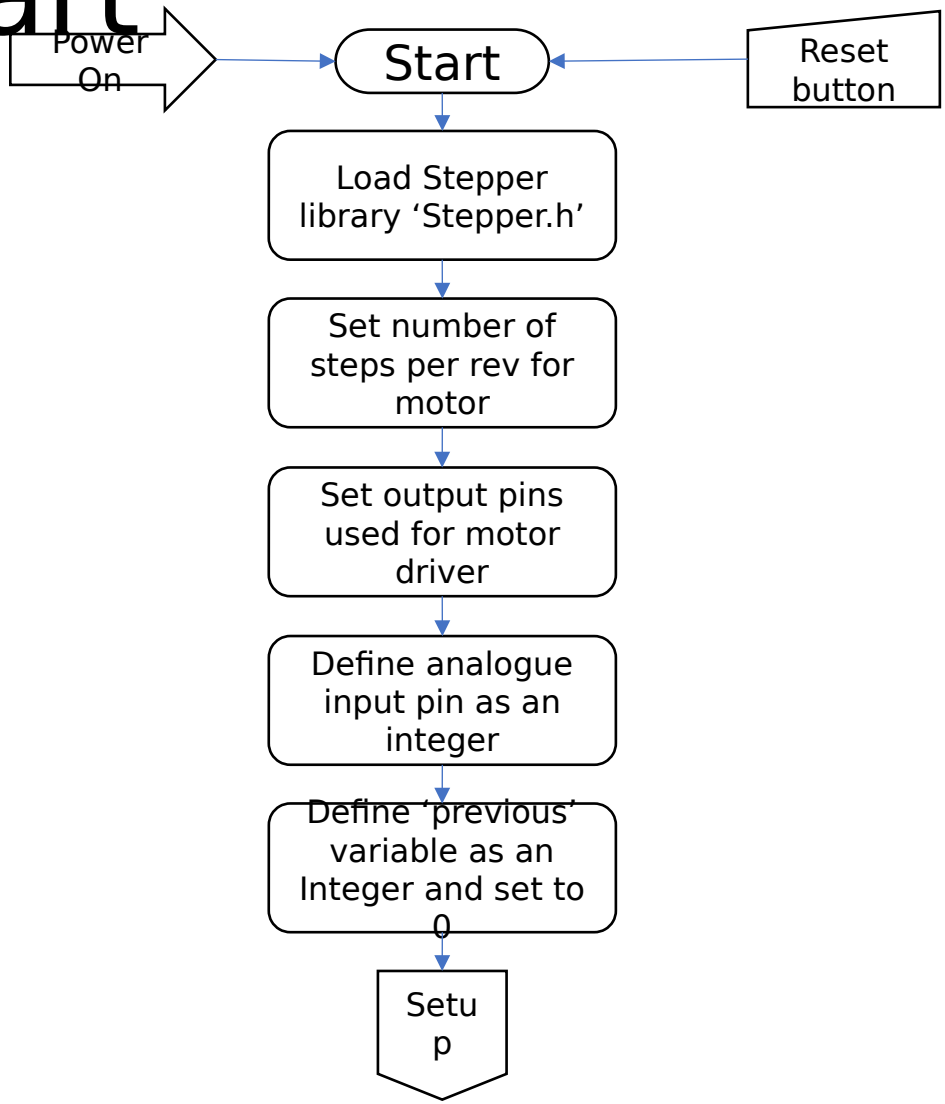
Example program

- Hardware
  - Stepper motor with driver board.
  - 10KΩ potentiometer connected to analogue input A0.
  - Arduino Leonardo board.
- Object of program
  - To make the stepper motor follow the position of the potentiometer.

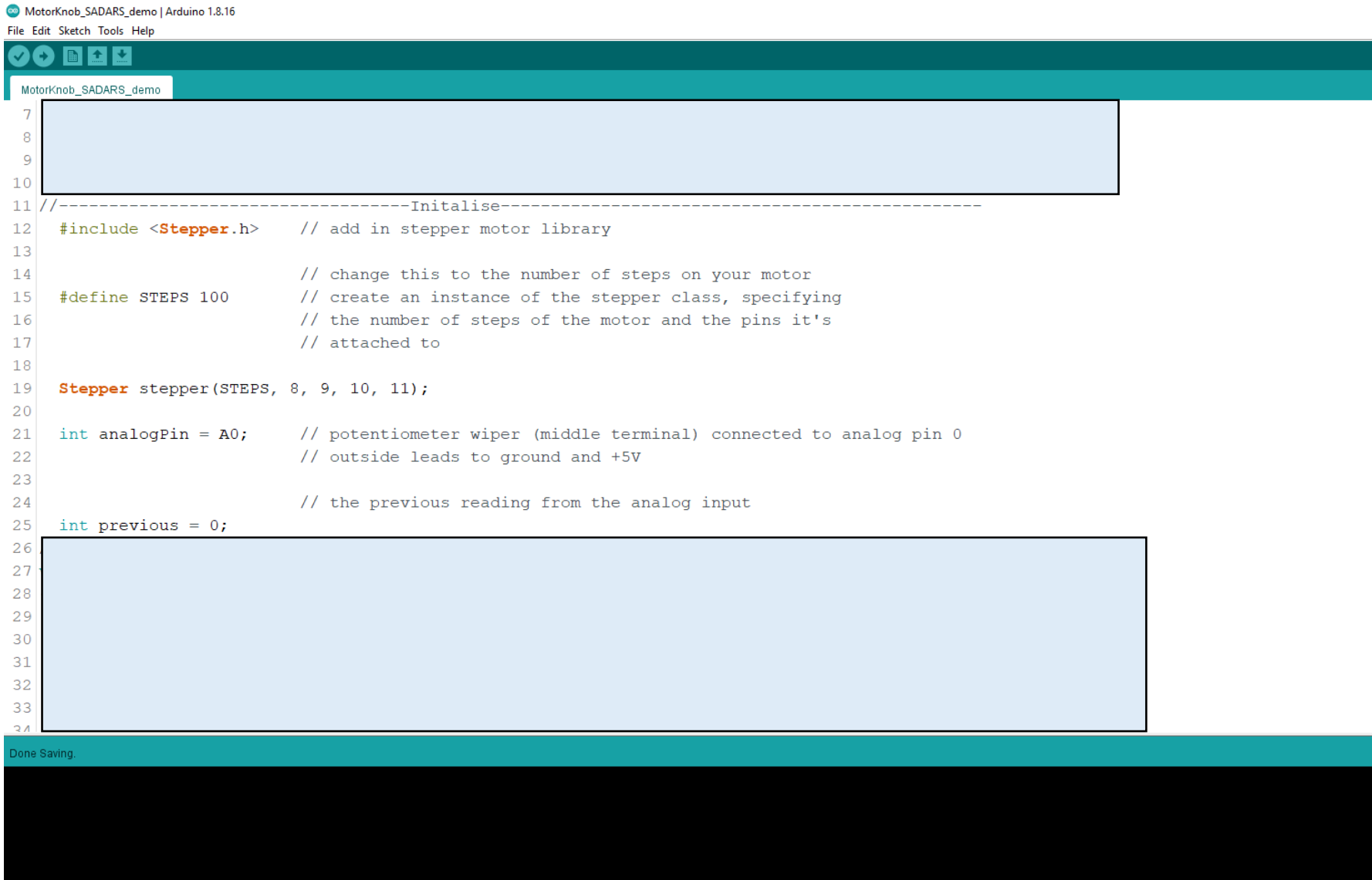
Circuit diagram



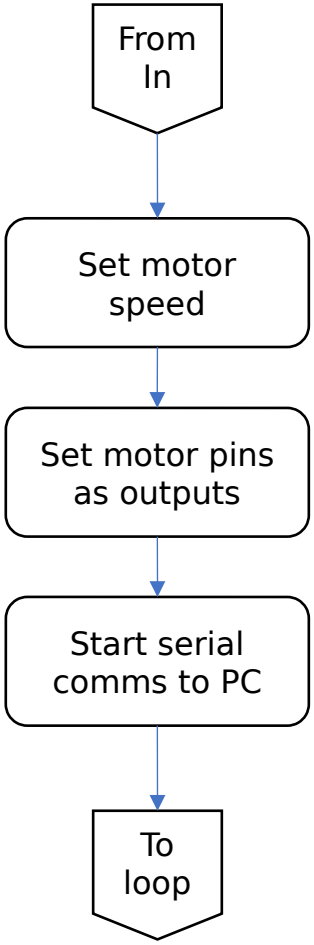
Example Initialise flow chart



Initialisation



Example Setup flow chart



Setup

MotorKnob_SADARS_demo | Arduino 1.8.16

File Edit Sketch Tools Help

MotorKnob_SADARS_demo

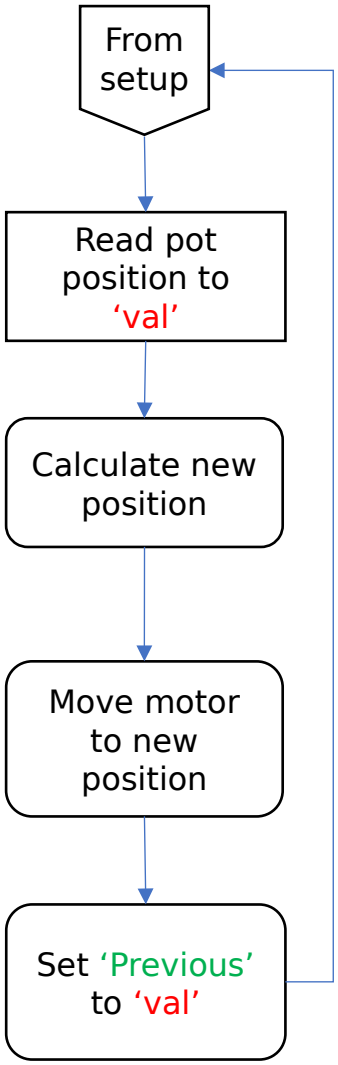
```
19
20
21
22
23
24
25
26 //----------------------------------Setup-----------------------------------------------------
27 void setup()
28 {
29
30   stepper.setSpeed(30);     // set the speed of the motor to 30 RPMs
31
32   pinMode(8, OUTPUT);       //set pin 8 as an output
33   pinMode(9, OUTPUT);       //set pin 9 as an output
34   pinMode(10, OUTPUT);      //set pin 10 as an output
35   pinMode(11, OUTPUT);      //set pin 11 as an output
36
37   Serial.begin(9600);       //start communication to pc
38 }
39
40
41
42
43
44
45   //Serial.println();
46
```

Done Saving.

Example loop flow chart

Following variable established in the initialisation and setup sections

- val
- Previous
  - Initially set to 0



Read the value of analogue input [A0] and store it in a variable called 'val'

Calculate the new position from the following 'val' – 'previous'

Move the stepper motor by the calculated difference

Set the current position into the variable called 'previous'

Loop

MotorKnob_SADARS_demo | Arduino 1.8.16

File Edit Sketch Tools Help

MotorKnob_SADARS_demo

```
//---------------------------------Program loop-----------------------------------------------
void loop()
{

  int val = analogRead(analogPin); // get the sensor value
  Serial.println(val);             // send input value to PC
  //Serial.println();

  stepper.step(val - previous);    // move a number of steps equal to the change in the
                                   // sensor reading

  previous = val;                  // remember the previous value of the sensor
}
//---------------------------------------------------------------------------------------------------------------------------
```

Done Saving.

25

Type here to search

11°C Partly sunny

Demonstration

- Show program in the Arduino IDE
- Upload into the Arduino Uno
- Run the program
- Demonstrate the changes in the Potentiometer cause the required changes in the motor position.