Sparkfun Electronics ATtiny85 Arduino Quick Reference Sheet
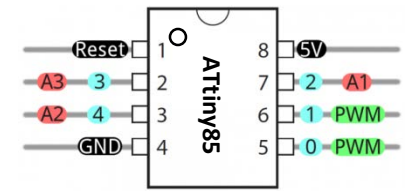

STRUCTURE

```
/* Each Arduino sketch must contain the
following two functions. */


void setup()
{ // this code runs once at the
  // beginning of the code execution.
}


void loop()
{ // this code runs repeatedly over
  // and over as long as the board is
  // powered.
}
```

COMMENTS

```
// this is a single line comment
/* this is
 a multiline
 comment */
```

SETUP

```
pinMode(pinNum, INPUT/OUTPUT/INPUT_PULLUP);
/* Sets the mode of the digital I/O pin.
All pins are general I/O on the board. You
must define what the pin will be used for at
the beginning of your code in setup() */
```

CONTROL STRUCTURES

```
if(condition)
{ // if condition is true, do something here
}
else
{ // otherwise, do this
}

for(init; condition; increment)
{
  // do this, increment, and
  // repeat while condition is true.
}
```

DIGITAL I/O

```
digitalWrite(pin, val);
```

```
/* val = HIGH or LOW write a HIGH or a LOW
value to a digital pin. */


buttonVal = digitalRead(pin);
```

```
/* Reads the value from a specified digital
pin, either HIGH or LOW. */
```

ANALOG I/O

```
analogWrite(pin, val);
```

```
/* Writes an analog voltage (using PWM) to a
pin. val = integer value from 0 to 255 */


sensorVal = analogRead(pin);
```

```
/* Reads the voltage from the specified
analog pin. 0V returns 0; Vcc returns 1023*/
```

TIME

```
delay(time_ms);
/* Pauses the program for the amount of time
(in milliseconds). */

millis();
/* Returns the number of milliseconds since
the board began running the current program.
max: 4,294,967,295 */
```
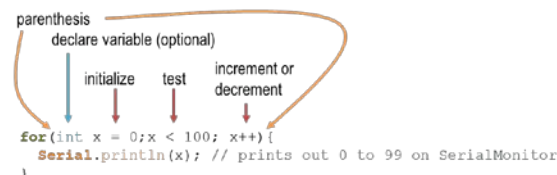
ATtiny85 Pins

Pins 0 – 4 : general purpose I/O pins (GPIO).

Both **digitalWrite()** and **digitalRead()** can be used with any of these pins.

Pins 0 & 1 : setup for PWM output using **analogWrite()**.

Pins A1, A2, A3 : setup for reading sensor input with **analogRead()**.

DATA TYPES

```
void      // nothing is returned
boolean  // 0, 1, false, true
char     // 8 bits: -128 to 127
byte     // 8 bits: 0 to 255
int      // 16 bits: -32,768 to 32,767
unsigned int   // 16 bits (unsigned)
long    /* 32 bits: -2,147,483,648
  to 2,147,483,647 */
unsigned long  // 32 bits (unsigned)
float   // 32 bits, signed decimal
```

SERIAL COMMUNICATION

A separate USB to serial adapter like FTDI is needed for Serial communication with the ATtiny. And. the ATtiny must be flashed to run at 8 MHz instead of 1 MHz.

The ATtiny does not support Serial natively. You need to use the **SoftwareSerial** library to enable this function.


```
for(int x = 0;x < 100; x++){
  Serial.println(x); // prints out 0 to 99 on SerialMonitor
}
```

```
#include <SoftwareSerial.h> // include library

SoftwareSerial tinySerial(3, 4);
/* Put above setup() and loop() - declares
tinySerial using 3 & 4 for Transmit (tx) and
Receive (rx) */

tinySerial.begin(9600); /* begin Serial at 9600
baud. Put this line in setup() */

tinySerial.print(""); /* sends data on TX line
- to your receiving computer. */

tinySerial.println(""); /* sends data to Serial
Monitor with CRLF. */

inChar = tinySerial.read();
```