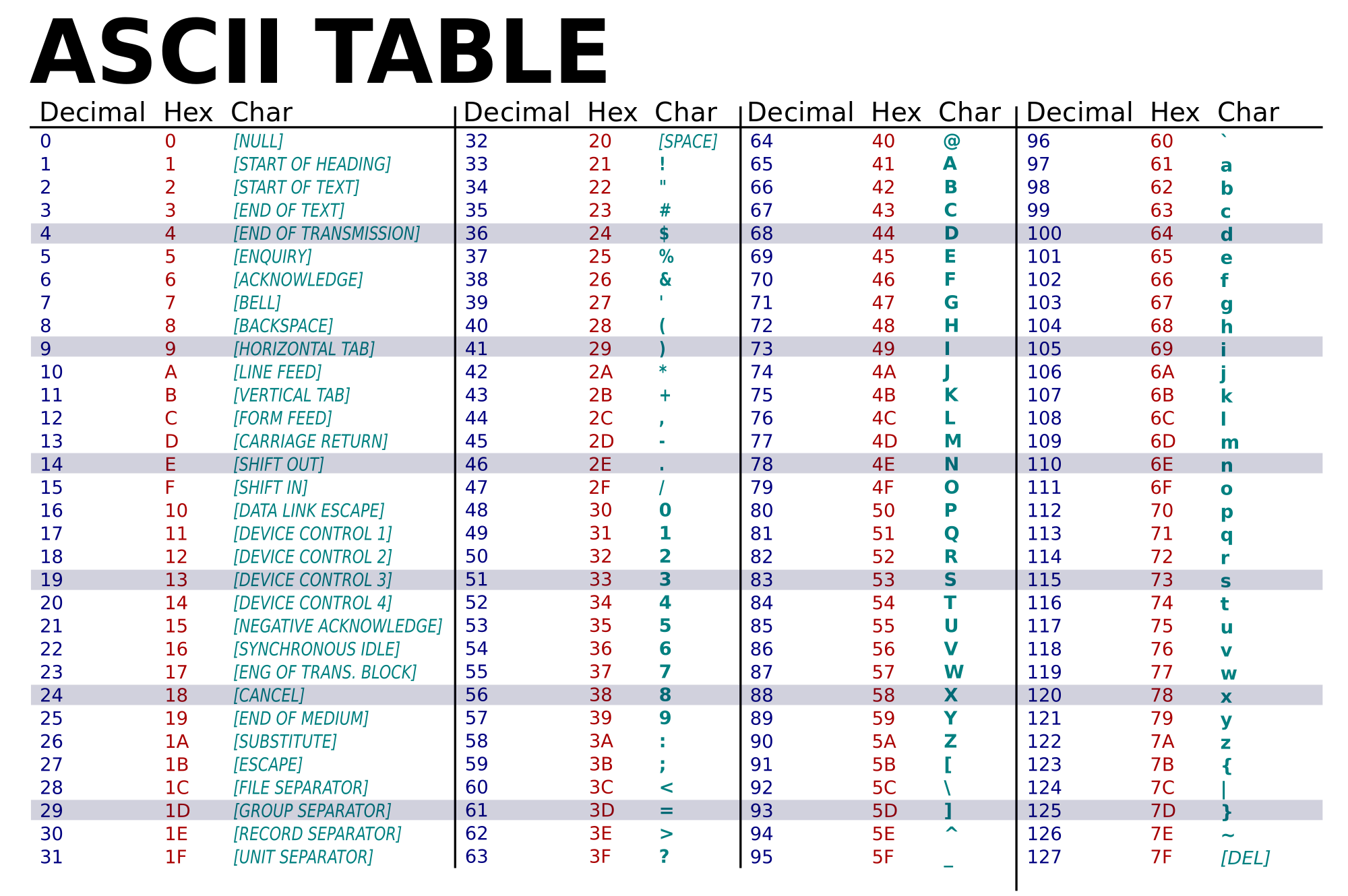
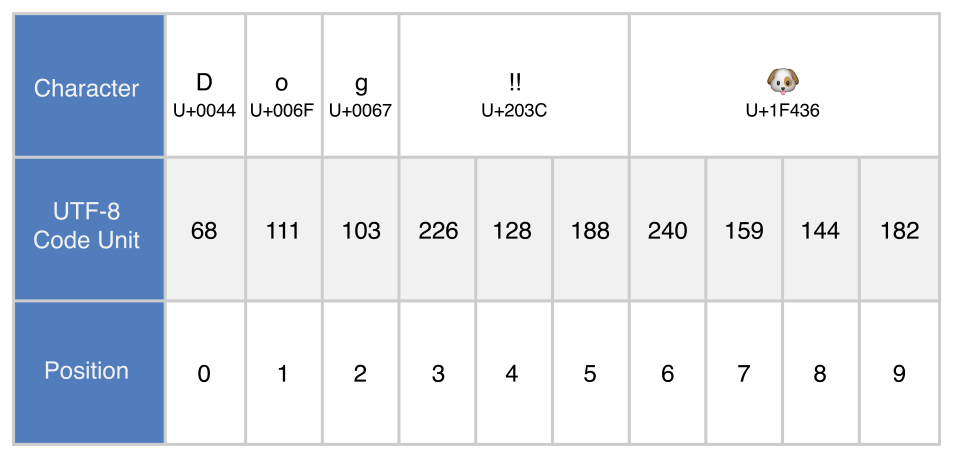
Devices and IOT Reference Materials for 11/10/16 class

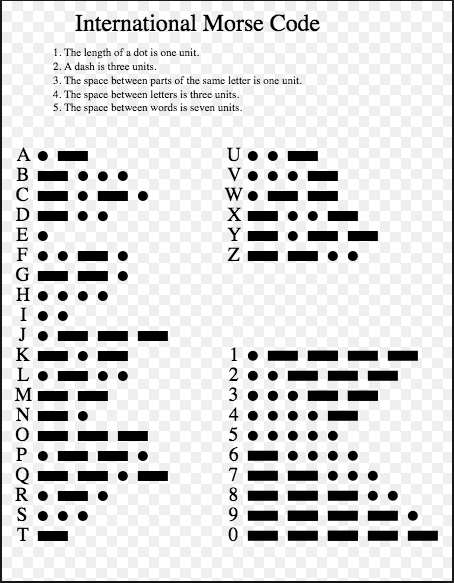


Unicode UTF-8

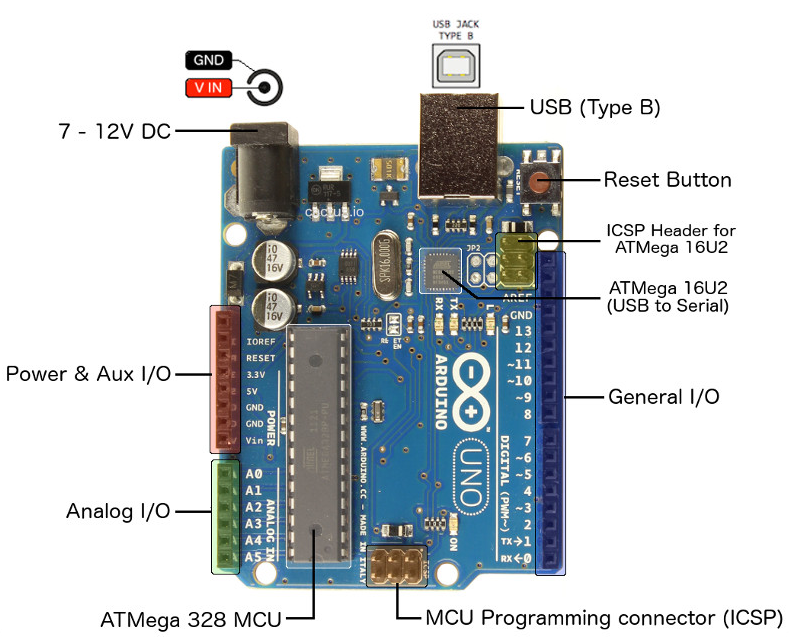




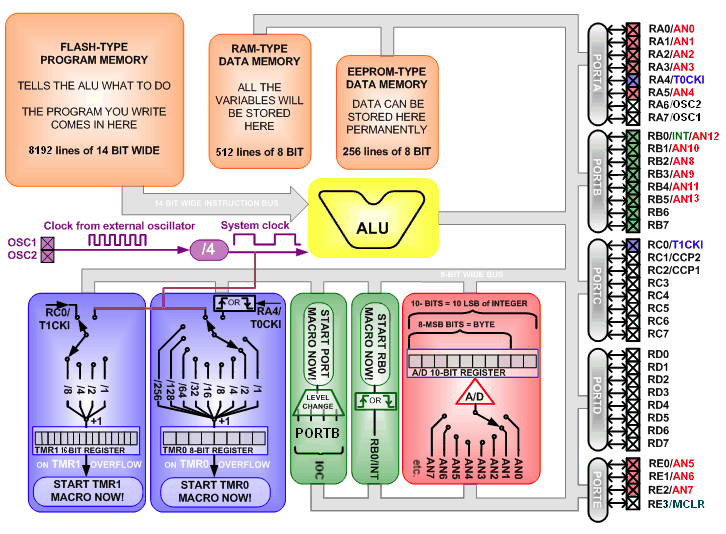




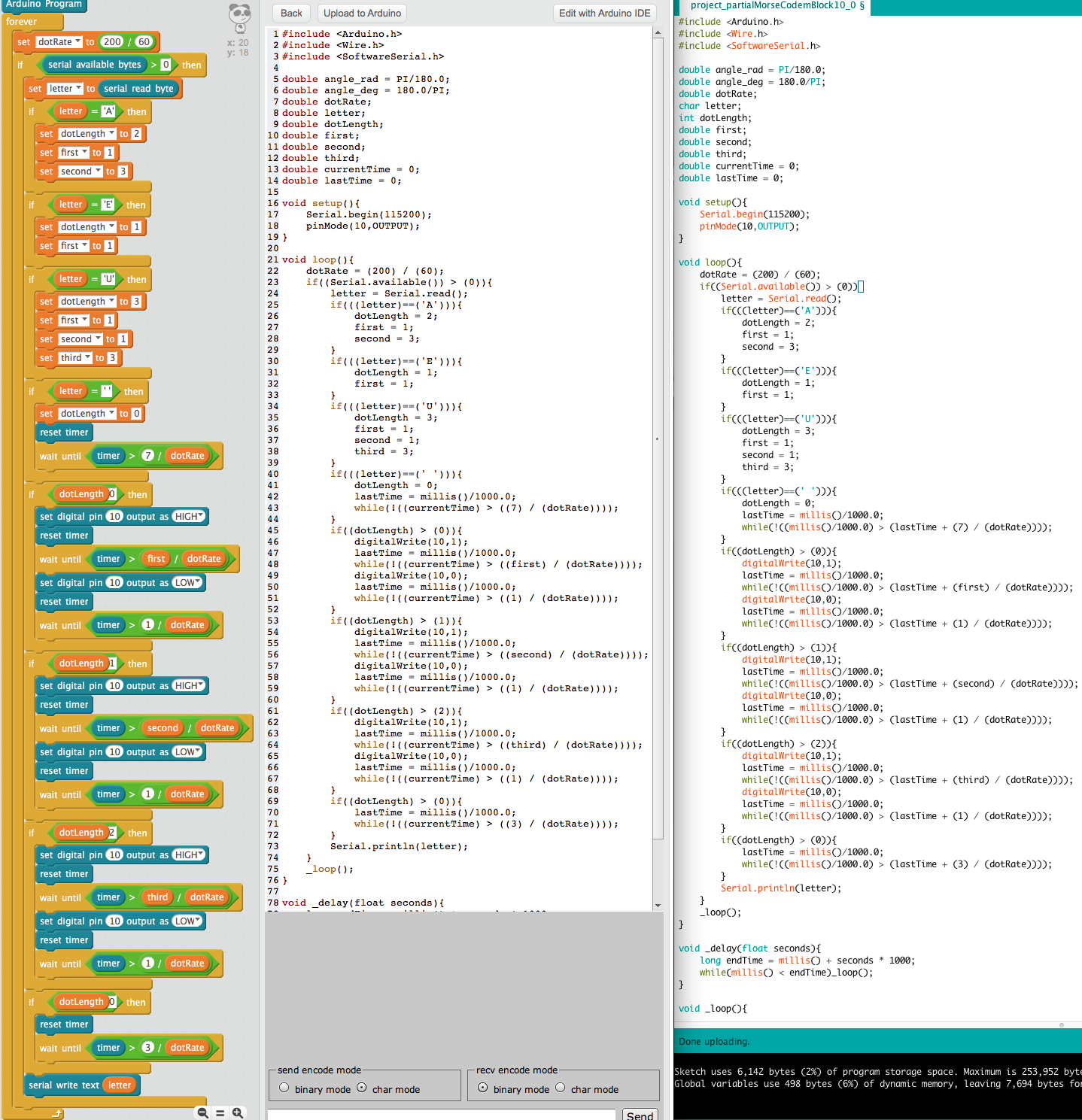
Arduino Uno components and interface pins and connector Arduino MEGA2560

Typical microcontroller “harvard” cpu architecture, with selectable clocks, timers, multiplexed input A/D converter and digital interface pins organized as banks of 8 pins



mBlock partial program to encode Morse Code letters ‘A’, ‘E’ and ‘U’ as blinks on an LED



Note: Unfortunately mBlock code generation has a bug in its implementation of the “wait until” block. See the two code sections and compare the use of uninitialized “currentTime” variable and a failure to include the “lastTime” variable correctly.

Please have a working version of the Morse Code encoder working by Tuesday morning start of class 11/15/16

This is the skeleton version of the file requiring you to code several sections as described in the code comments. Look up bitwise AND and shift right operators & and >> if they are unfamiliar.

------------------------

// Morse code encoder project for devices and IOT class

// Skeleton coded by Shannon Bailey, 11/9/2016

#include <Arduino.h>

double dotRate = (200) / (60);

char letter;

double lastTime = 0;

int letterIndex;

int letterLength;

int ditsDahs;

int count;

int ditOrDah;

// A table of MorseCode letters are coded as the length of each letter, and in

// in dits and dahs which are coded as 0 bits = dits and 1 bits = dahs

// There are 8 Bits in a Byte and like decimal numbers have thousands, hundreds, tens and ones,

// binary number bits are powers of 2 which are 128, 64, 32, 16, 8, 4, 2, 1

// for example: Morse code letter 'C' is length 4 and dah, dit, dah, dit or 1010 in binary and

// binary 1010 has a 1 in the 4th bit and 2nd bit so its value is 8 + 2 = 10 in decimal.

char MorseCodeLetters[] = {"ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"};

// A B C D E F G H I J K L M N O P Q R S T U V W X Y Z 1 2 3 4 5 6 7 8 9 0

int MorseCodeLetterLengths[] = { 2, 4, 4, 3, 1, 4, 3, 4, 2, 4, 3, 4, 2, 2, 3, 4, 4, 3, 3, 1, 3, 4, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5 };

int MorseCodeLetterDitsDahs[] = {0x01, 0x1000, 0x1010, 0x100, 0x0, 0x0010, 0x110, 0x0000, 0x00, 0x0111, 0x101, 0x0100, 0x11, 0x10, 0x111, 0x0110, 0x1101, 0x010, 0x000, 0x1, 0x001, 0x0001, 0x011, 0x1001, 0x1011, 0x1100, 0x01111, 0x00111, 0x00011, 0x00001, 0x00000, 0x10000, 0x11000, 0x11100, 0x11110, 0x11111};

//int MorseCodeLetterDitsDahs[] ={ 1, 8, 10, 4, 0, 2, 6, 0, 0, 7, 5, 4, 3, 2, 7, 6, 13, 2, 0, 1, 1, 1, 3, 9, 11, 12, 15, 7, 3, 1, 0, 16, 24, 28, 30, 31 };

void setup(){

Serial.begin(115200);

pinMode(10, OUTPUT);

}

void loop() {

if((Serial.available()) > (0)){

letter = toupper( Serial.read() );

Serial.write(letter);

// check for space character and wait for 7 dot time slots

if (letter == ' '){

lastTime = millis()/1000.0;

while(!((millis()/1000.0) > (lastTime + (7) / (dotRate))));

} else {

// You can use a for loop and if statement comparing each letter to MorseCodeLetters[letterIndex] to find the letterIndex

{

// note: a faster solution would use if statements to check the range A-Z and 0-9 and calculate the letterIndex

{

letterLength = MorseCodeLetterLengths[ letterIndex ];

ditsDahs = MorseCodeLetterDitsDahs[ letterIndex ];

// use a for loop to iterate through each bit

{

// shift and isolate each bit and set its to 1 if a dit, and 3 if a dah

// use the right shift operator >> use the bitwise (binary) AND operator & and ternary (true/false) ? trueValue : falseValue

// be careful to compute the amount to shift correctly starting with the left most bit, so for a length 4, shift 3, 2, 1, 0

// ditOrDah = ? ;

digitalWrite(10, 1);

// wait for dit or dah time

digitalWrite(10, 0);

// wait for 1 dit time

}

// after each letter, add a 3 dit time wait between each letter

// do you need to break out of the for loop?

}

}

}

}

}

**Reference material:**

<https://commons.wikimedia.org/wiki/File:International_Morse_Code.svg>

Submarine flashing morse code

<https://www.youtube.com/watch?v=knAQXAFvQik>

Vibraplex high speed Morse code key

<https://www.youtube.com/watch?v=yHz2rEiFnfw>

Word per minute equals 2.4 times the length of a dot

<http://www.nu-ware.com/NuCode%20Help/index.html?morse_code_structure_and_timing_.htm>

**Further Resources:**

Energy harvesting

<http://core.spansion.com/article/energy-harvesting-devices-replace-batteries-in-iot-sensors/#.WB0jYuErKEI>