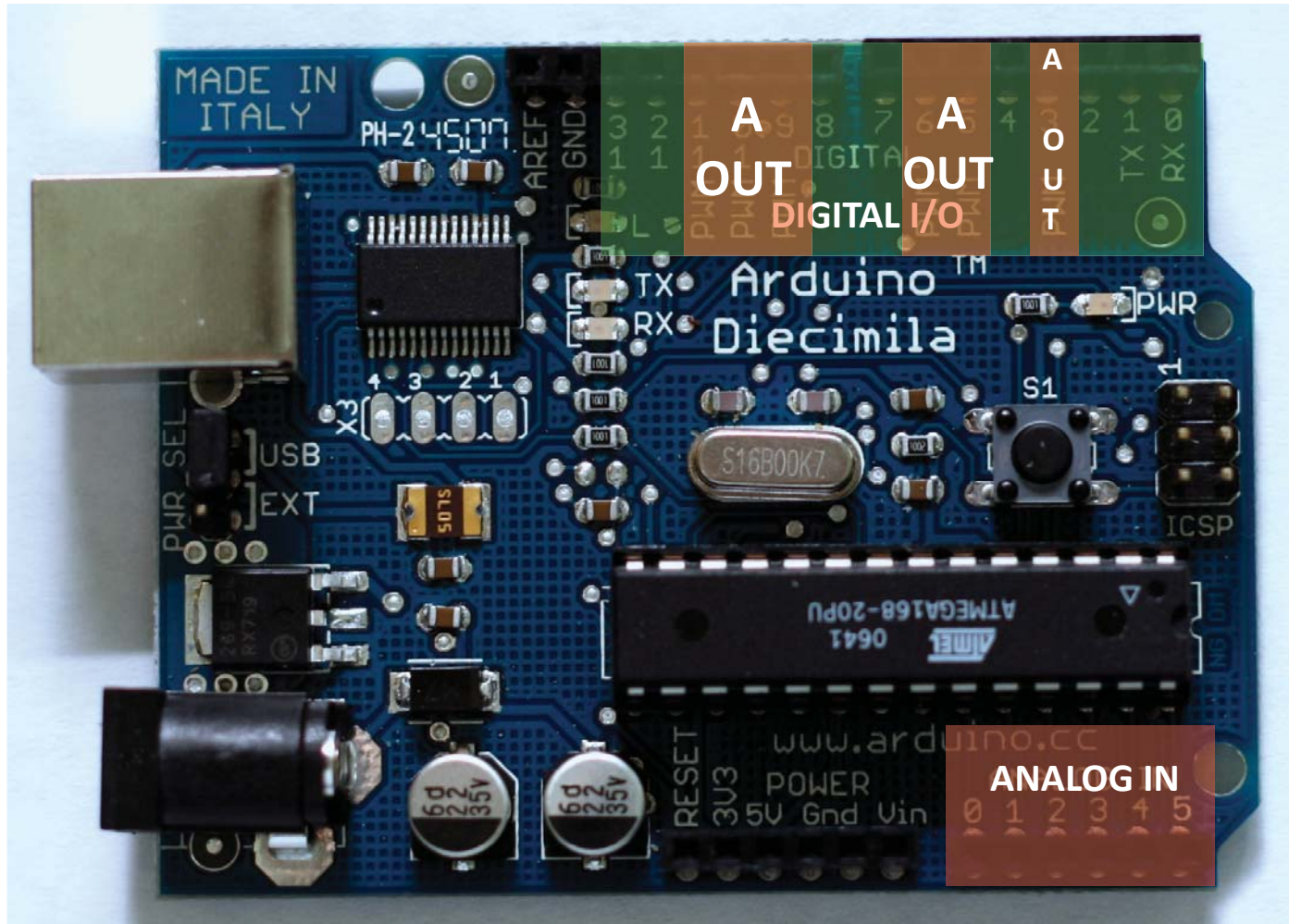


DIGITAL & ANALOG I/O_QUIK RECAP

DIGITAL AND ANALOG I/O ON THE BOARD



Digital In:
High / Low

Digital Out:
High / Low

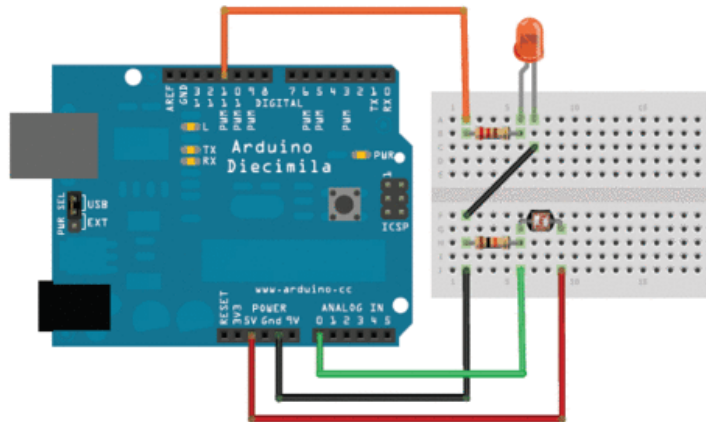
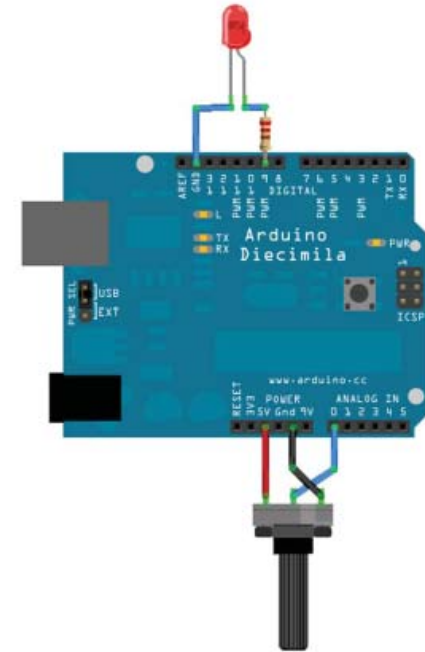
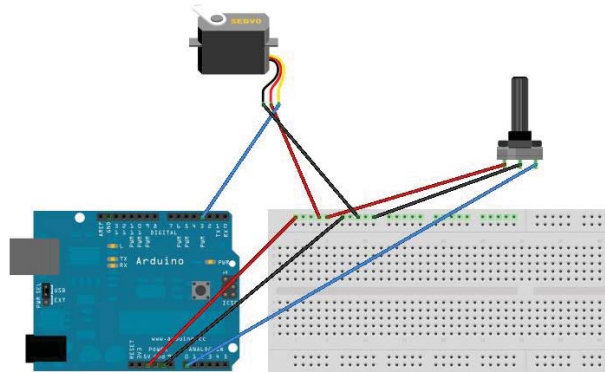
Analog In:
0 – 1023

Analog Out:
0 – 255

RECAP

ADC
PWM

Potentiometer
LED dimming
Piezo Speaker
Servo Motor



CODING IN ARDUINO

Brief introduction:

- on the project, references, types of electronics used, Arduino pins used etc.

Include external library:

- `#include <filename.h>`

Declaring Variables:

- Introducing all different numeric, character types used in the program

Structure:

- Setup Function:

- Start of Sketch, One time Run
- Initialization of variables, assigning pins etc.

- Loop Function:

- Keeps looping the program written in this function
- Main coding area
- Custom Functions:

- Write our own functions

COMMENTING

/*

Basic Intro to project (function / usefulness)

Type of electronics used in project

Considerations / Assumptions / Cautions /

Limitations

Creation date / Author

References

*/

// Single line description of the current line in the code

LIBRARIES

#include <filename.h> //imports outside code into code
To load external libraries (set of functions)

<http://www.arduino.cc/playground/ComponentLib/Servo>

```
#include <Servo.h>
```

```
Servo myservo;      //create servo object to control a servo
int potpin = 0;      // analog pin used to connect the potentiometer
int val = 0;         // variable to read the value from the analog pin
void setup()
{
    myservo.attach(2); // attaches the servo on pin 2 to the servo object
}
void loop()
{
    val = analogRead(potpin);    // reads the value of the potentiometer (value between 0 and
1023)
    val = map(val, 0, 1023, 0, 180); // scale it to use it with the servo (value between 0 and 180)
    myservo.write(val);          // sets the servo position according to the scaled value
    delay(15);                   // waits for the servo to get there
}
```

All Arduino libraries are in the following folder :

\arduino-0017\arduino-0017\hardware\libraries

Add new Header files: copy files in the same folder

#define = #include

DECLARING VARIABLES

- Storage packets of values
- Types:
 - Constants
 - HIGH / LOW
 - INPUT / OUPUT
 - TRUE / FALSE
 - Data Types
 - Conversion

int variable; int variable = value;

boolean: true or false values

declaration boolean x = true; x = !x;

char: to assign Decimal ASCII character

declaration: char letter = 'A'; char letter = 65;

byte: 8-bit number

declaration: byte b = B10010; // (B10010 = 18 decimal)

int: integers (-32768 to 32767) (-2^{15} to $2^{15}-1$)

delaration: int degrees = -365;

unsigned int: integers (0 to 65535)

delaration: unsigned int days = 365;

long: extended variable size for integers (-2147483648 to 2147483647)

declaration: long val = -2147483647;

unsigned long: 0 to $2^{32}-1$

declaration: unsigned long val = 2147483647;

float /double: takes decimal values ($-\pi$ to π with 38 decimal digits)

declaration: float pi = 3.14;

array: collection of variables accessed with index number

declaration: int pins[6];

int pins[] = {2, 4, 8, 3, 6};

int pins[6] = {2, 4, -8, 3, 2};

string: arrays of char

declaration: char Str3[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o', '\0'};

char Str4[] = "arduino";

void: used for function declaration

declaration: void name(_)

String(object): allows you to use and manipulate strings of text in more complex ways than character arrays do

String(object) declarations:

// using a constant String

String stringOne = "Hello String";

// concatenating two strings

String stringOne = String(stringTwo + "with more");

// using a constant integer

String stringOne = String(13);

// using an int and a base

String stringOne = String(analogRead(0), DEC);

// using an int and a base (hexadecimal)

String stringOne = String(45, HEX);

// using an int and a base (binary)

String stringOne = String(255, BIN);

// using a long and a base

String stringOne = String(millis(), DEC);

Ctrl	Dec	Hex	Char	Code	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
^@	0	00		NUL	32	20	sp	64	40	@	96	60	`
^A	1	01	␣	SOH	33	21	!	65	41	A	97	61	a
^B	2	02	␢	SIX	34	22	"	66	42	B	98	62	b
^C	3	03	♥	EIX	35	23	#	67	43	C	99	63	c
^D	4	04	♦	EOI	36	24	\$	68	44	D	100	64	d
^E	5	05	♣	ENQ	37	25	%	69	45	E	101	65	e
^F	6	06	♠	ACK	38	26	&	70	46	F	102	66	f
^G	7	07	•	BEL	39	27	'	71	47	G	103	67	g
^H	8	08	◼	BS	40	28	(	72	48	H	104	68	h
^I	9	09	○	HI	41	29	)	73	49	I	105	69	i
^J	10	0A	◻	LF	42	2A	*	74	4A	J	106	6A	j
^K	11	0B	♂	VI	43	2B	+	75	4B	K	107	6B	k
^L	12	0C	♀	FF	44	2C	,	76	4C	L	108	6C	l
^M	13	0D	␣	CR	45	2D	-	77	4D	M	109	6D	m
^N	14	0E	␣	SD	46	2E	.	78	4E	N	110	6E	n
^O	15	0F	✱	SI	47	2F	/	79	4F	O	111	6F	o
^P	16	10	▶	SLE	48	30	0	80	50	P	112	70	p
^Q	17	11	◀	CS1	49	31	1	81	51	Q	113	71	q
^R	18	12	↕	DC2	50	32	2	82	52	R	114	72	r
^S	19	13	!!	DC3	51	33	3	83	53	S	115	73	s
^T	20	14	¶	DC4	52	34	4	84	54	T	116	74	t
^U	21	15	§	NAK	53	35	5	85	55	U	117	75	u
^V	22	16	▬	SYN	54	36	6	86	56	V	118	76	v
^W	23	17	⌘	EIB	55	37	7	87	57	W	119	77	w
^X	24	18	↑	CAN	56	38	8	88	58	X	120	78	x
^Y	25	19	↓	EM	57	39	9	89	59	Y	121	79	y
^Z	26	1A	→	SIB	58	3A	:	90	5A	Z	122	7A	z
^[	27	1B	←	ESC	59	3B	;	91	5B	[	123	7B	{
^\	28	1C	⌞	FS	60	3C	<	92	5C	\	124	7C	
^]	29	1D	↗	GS	61	3D	=	93	5D	]	125	7D	}
^^	30	1E	▲	RS	62	3E	>	94	5E	^	126	7E	~
^_	31	1F	▼	US	63	3F	?	95	5F	_	127	7F	Δ [†]

† ASCII code 127 has the code DEL. Under MS-DOS, this code has the same effect as ASCII 8 (BS). The DEL code can be generated by the CTRL+BKSP key.

STRUCTURE

- Setup() & Loop()
- Control Structures:
 - If / if... else
 - For
 - Switch case
 - While / do ... while
 - Break
 - Continue
 - return

STRUCTURE

```
int buttonState =0;
unsigned long count = 0;
void setup()
{
  Serial.begin(9600);
}
void loop()
{
  /*buttonState = digitalRead(2);
  while(buttonState == HIGH)          //infinite loop
  {
    count = (count+1);
  }*/
  while(digitalRead(2) == HIGH)
  {
    count = (count+1);
    if((count/1000) > 600)           //stop counting when count reach 600
    {
      break;
    }
  }
  if(count>0)
  {
    Serial.println(count/1000); //convert milliseconds to seconds
  }
  count = 0;
}
```

STRUCTURE

Functions

- System functions (<http://arduino.cc/en/Reference/HomePage>)
 - Digital I/O
 - Analog I/O
 - Advanced I/O
 - Time
 - Math
 - Trigonometry
 - Random Numbers
 - Communication
- Custom functions (<http://www.arduino.cc/en/Reference/FunctionDeclaration>)

CUSTOM FUCTIONS

```
int a = 9;  
int b = 6;  
int c = 0;  
  
void pythagorean(int a, int b)  
{  
    c = sqrt((a^2) + (b^2));  
}
```

The diagram illustrates the flow of data between a main program and a function. It shows three integer variables in the main scope: `a = 9`, `b = 6`, and `c = 0`. Below them is a function definition `void pythagorean(int a, int b)` which calculates the hypotenuse `c` using the Pythagorean theorem. Two curved arrows originate from the initial values of `a` and `b` and point to the corresponding parameters in the function signature. Inside the function, a third arrow points from the local variable `c` back to the global variable `c` in the main scope, indicating that the function updates the global variable.