Energia Cheat Sheet V.01c

By F4DTR (Jean-Yves) – 2013 – HackGyver, French Hackspace (Belfort).

Based on Arduino Cheat Sheet V0.2c by Gavin Smith – Robots and Dinosaurs, The Sydney Hackspace & Rei Vilo - embeddedcomputing.weebly.com

Reference :

Structure :
```
void setup()   void loop()
```

Control Structures :
```
if (x<5){  } else {  }
switch (myvar) {
   case 1:
     break;
   case 2:
     break;
   default:
}
for (int i=0; i <= 255; i++){   }
while (x<5){  }
do {  } while (x<5);
continue; //Go to next in do/for/while loop
return x; // Or 'return;' for voids.
goto          // considered harmful :-)
```

General Operators :
```
= (assignment operator)
+  (addition)          - (subtraction)
* (multiplication)  / (division)
% (modulo)
== (equal to)        != (not equal to)
< (less than)        > (greater than)
<= (less than or equal to)
>= (greater than or equal to)
&& (and)      || (or)       ! (not)
```

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

Compound Operators :
```
++ (increment)     -- (decrement)
+= (compound addition)
-= (compound subtraction)
*= (compound multiplication)
/= (compound division)
&= (compound bitwise and)
|= (compound bitwise or)
```

Math :
```
min(x, y)     max(x, y)      abs(x)
constrain(x, minval, maxval )
map(val, fromL, fromH, toL, toH)
pow(base, exponent)  sqrt(x)
sin(rad)        cos(rad)           tan(rad)
```

Pointer Access :
```
& reference;  * dereference operator
```

Further Syntax :
```
// (single line comment)
/*  (multi-line comment) */
#define DOZEN 12 //Not baker's!
#include <avr/pgmspace.h>
```

Constants :
```
HIGH | LOW
INPUT | OUTPUT
true | false
143 // Decimal number
0173 // Octal number
0b11011111 //Binary
0x7B // Hex number
7U // Force unsigned
10L // Force long
15UL // Force long unsigned
10.0 // Forces floating point
2.4e5 // 240000
```

Data Types :
```
void
boolean            (0, 1, false, true)
char   (e.g. 'a' -128 to 127)
unsigned char (0 to 255)
byte    (0 to 255)
int       (-32,768 to 32,767)
unsigned int (0 to 65535)
word  (0 to 65535)
long          (-2,147,483,648 to
               2,147,483,647)
unsigned long (0 to 4,294,967,295)
float         (-3.4028235E+38 to
               3.4028235E+38)
double (currently same as float)
sizeof(myint) // returns 2 bytes

int, word, long can be replaced by :

[u]int{8|16|32|64}_t
[u] for unsigned, and {8|16|32|64} for size.
// it's better for portability
```

Strings :
```
char S1[15];
char S2[8]={'a','r','d','u','i','n','o'};
char S3[8]={'a','r','d','u','i','n','o','\0'};
    //Included \0 null termination
char S4[ ] = "arduino";
char S5[8] = "arduino";
char S6[15] = "arduino";
```

Qualifiers :
```
static  // persists between calls
volatile     // use  RAM (nice for ISR)
const  // make read-only
PROGMEM // use flash
```

Arrays :
```
int myInts[6];
int myPins[] = {2, 4, 8, 3, 6};
int mySensVals[6] = {2, 4, -8, 3, 2};
```

Bits and Bytes :
```
lowByte()             highByte()
bitRead(x,bitn)   bitWrite(x,bitn,bit)
bitSet(x,bitn)      bitClear(x,bitn)
bit(bitn) //bitn:  0-LSB 7-MSB
```

Conversion :
```
char()                  byte()
int()                    word()
long()                  float()
```

External Interrupts :
```
attachInterrupt(interrupt, function,
[LOW,CHANGE,RISING,FALLING])
detachInterrupt(interrupt)
interrupts()
noInterrupts()
```

Digital I/O :
```
pinMode(pin, [INPUT,OUTPUT])
digitalWrite(pin, value)
int digitalRead(pin)
//Write High to inputs to use pull-up res
```

Analog I/O :
```
analogReference([DEFAULT,INTERNA
L,EXTERNAL])
int analogRead(pin) //Call twice if
switching pins from high Z source.
analogWrite(pin, value) // PWM
```

Advanced I/O :
```
tone(pin, freqhz)
tone(pin, freqhz ,duration_ms)
noTone(pin)
shiftOut(dataPin, clockPin,
[MSBFIRST,LSBFIRST], value)
unsigned long pulseIn(pin, [HIGH,LOW])
```

Time :
```
unsigned long millis() // 50 days overflow.
unsigned long micros() // 70 min overflow
delay(ms)
delayMicroseconds(us)
```

Random Numbers :
```
randomSeed(seed) // Long or int
long random(max)
long random(min, max)
```

Libraries :

Stepper Motor (#include <Stepper.h>)
```
Stepper myStepper(nbStep, pinA,B,C,D);
.setSpeed(60); // 60 rpm
.step(100);  // mov. 100 stp Forw.
.step(-100); // mov. 100 stp Back.
```

Servo (#include <Servo.h>)
```
attach(pin , [min_uS, max_uS])
write(angle) // 0-180
writeMicroseconds(uS) //1000-2000, 1500 is
   midpoint
read() // 0-180
attached() //Returns boolean
detach()
```

Serial.
```
begin([300, 1200, 2400, 4800, 9600, 14400,
19200, 28800, 38400, 57600, 115200])
end()
int available()
int read()
flush()
print()
println()
write()
```

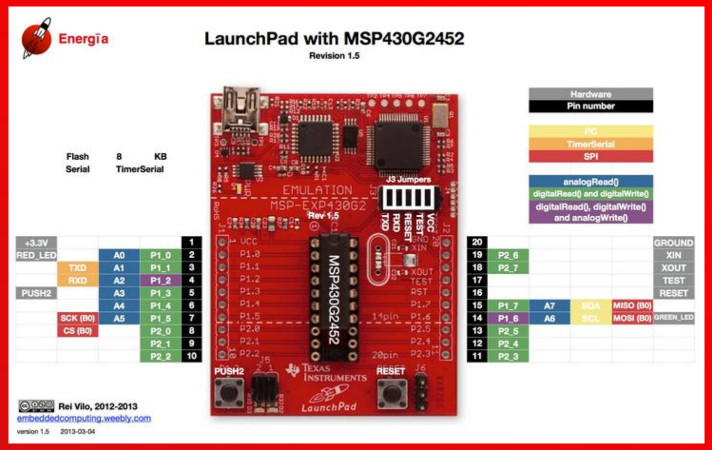
EEPROM (#include <EEPROM.h>)
```
byte read(intAddr) write(intAddr,myByte)
```

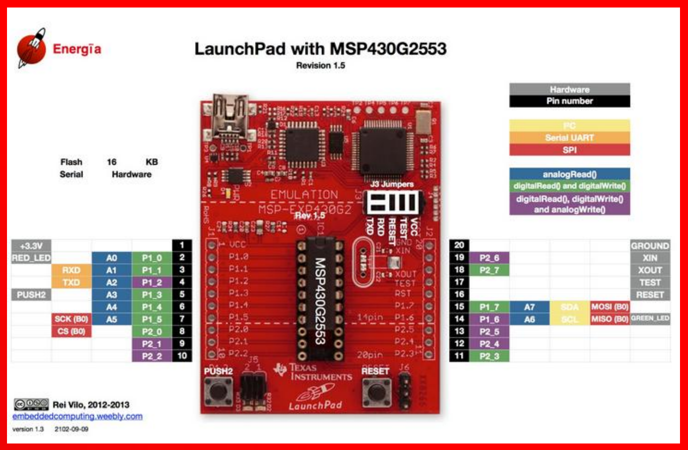
Wire (#include <TwoWire.h>) // or Wire.h, I2C Library.
```
begin()               // Join as master
begin(addr) // Join as slave @ addr
requestFrom(address, count)
beginTransmission(addr) // Step 1
write(mybyte)              // Step 2
write(byte * data, size)
endTransmission()          // Step 3
int available() // Num of bytes
int receive() //Return next byte
onReceive(handler)
onRequest(handler)
```

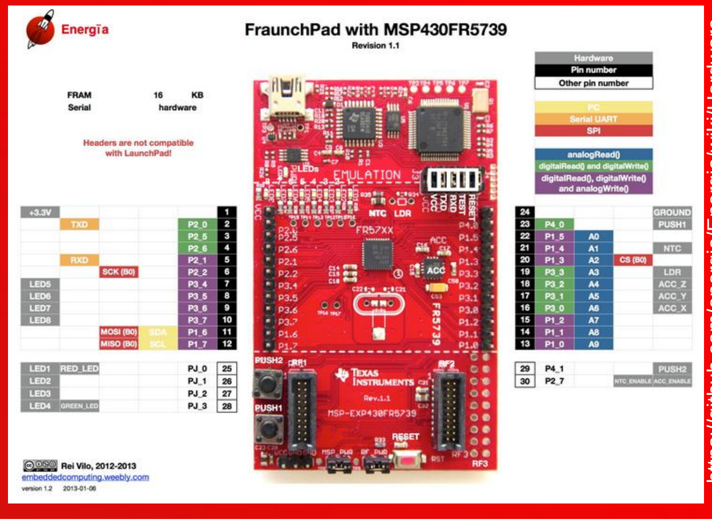
IN / OUT :

µC	Flash	SRAM	# of IO	Serial	Interrupt	PWM	Analog	SPI	I2C	USB	TEMP Sensor	RTC
2452	8	256	16	1	16	2	8	1	1	0	1	
2553	16	512	16	1	16	7	8	1	1	0	1	
FR5739	16	1024	32	1	32	14	10	1	1	0	1	1
LM4F120	256	32K	35	6	35	23	12	4	4	1	1	1

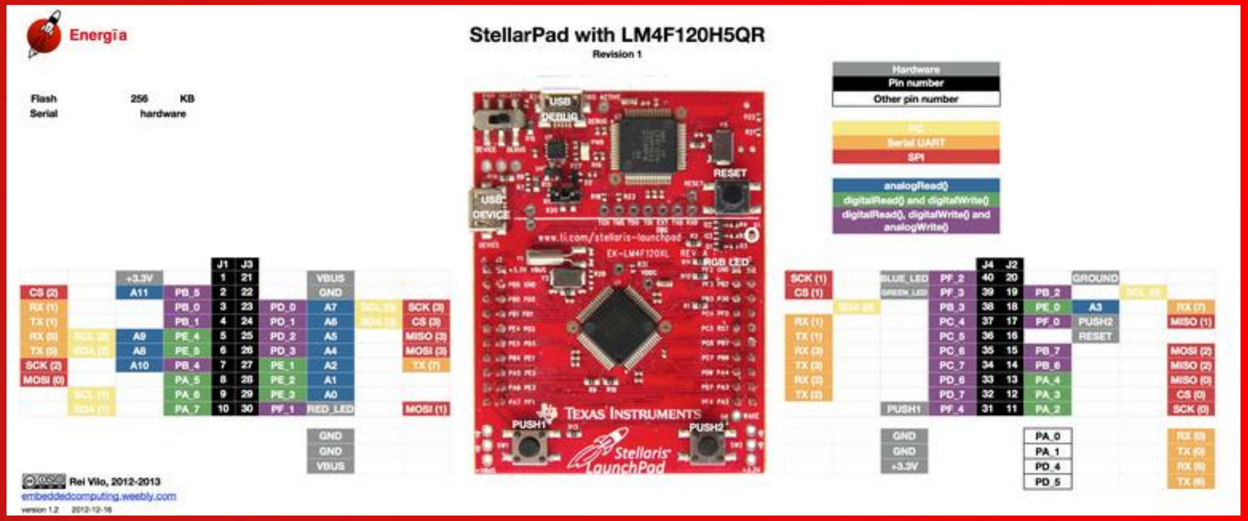
https://github.com/energia/Energia/wiki/Hardware
http://embeddedcomputing.weebly.com/launchpad-msp430.html






Energia
FraunchPad with MSP430FR5739
Revision 1.1
Headers are not compatible with LaunchPad!
Rei Vilo, 2012-2013
embeddedcomputing.weebly.com

https://github.com/energia/Energia/wiki/Hardware
http://embeddedcomputing.weebly.com



https://github.com/energia/Energia/wiki/Hardware
http://embeddedcomputing.weebly.com/stellarpad-lm4f.html