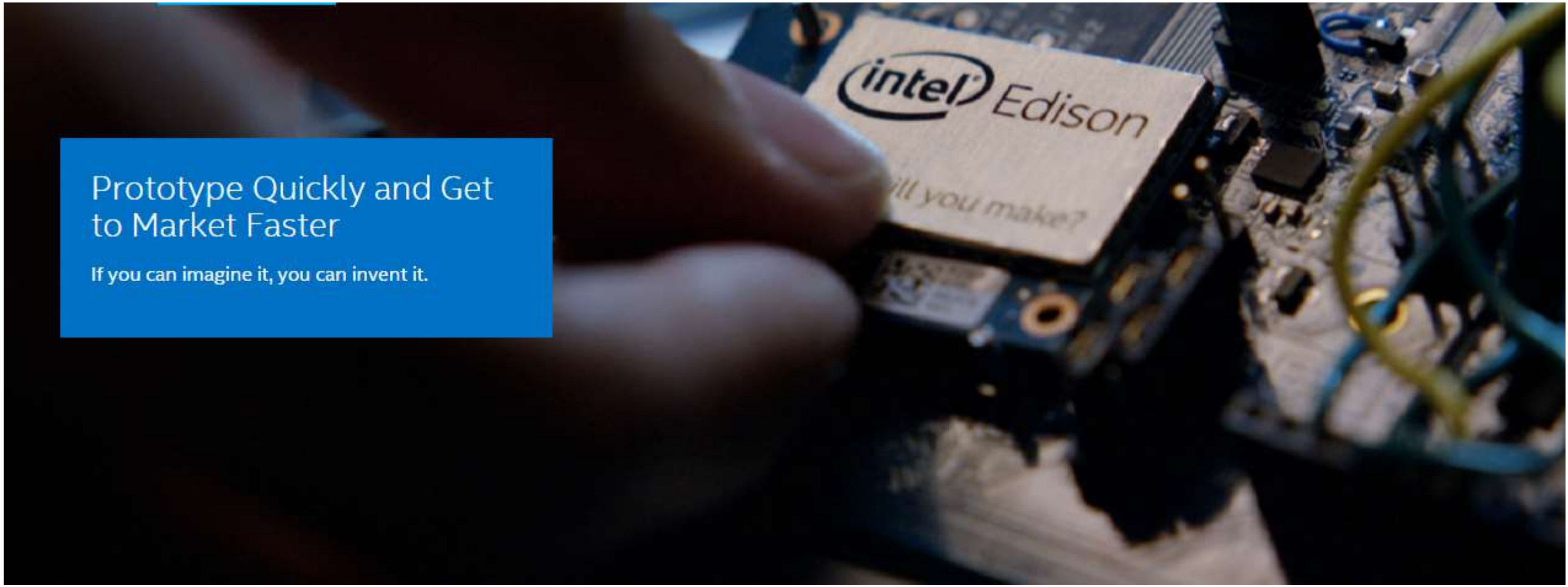
Making Exploring Robot with Edison

Intel Edison으로 만들어 보는 화성 탐사선

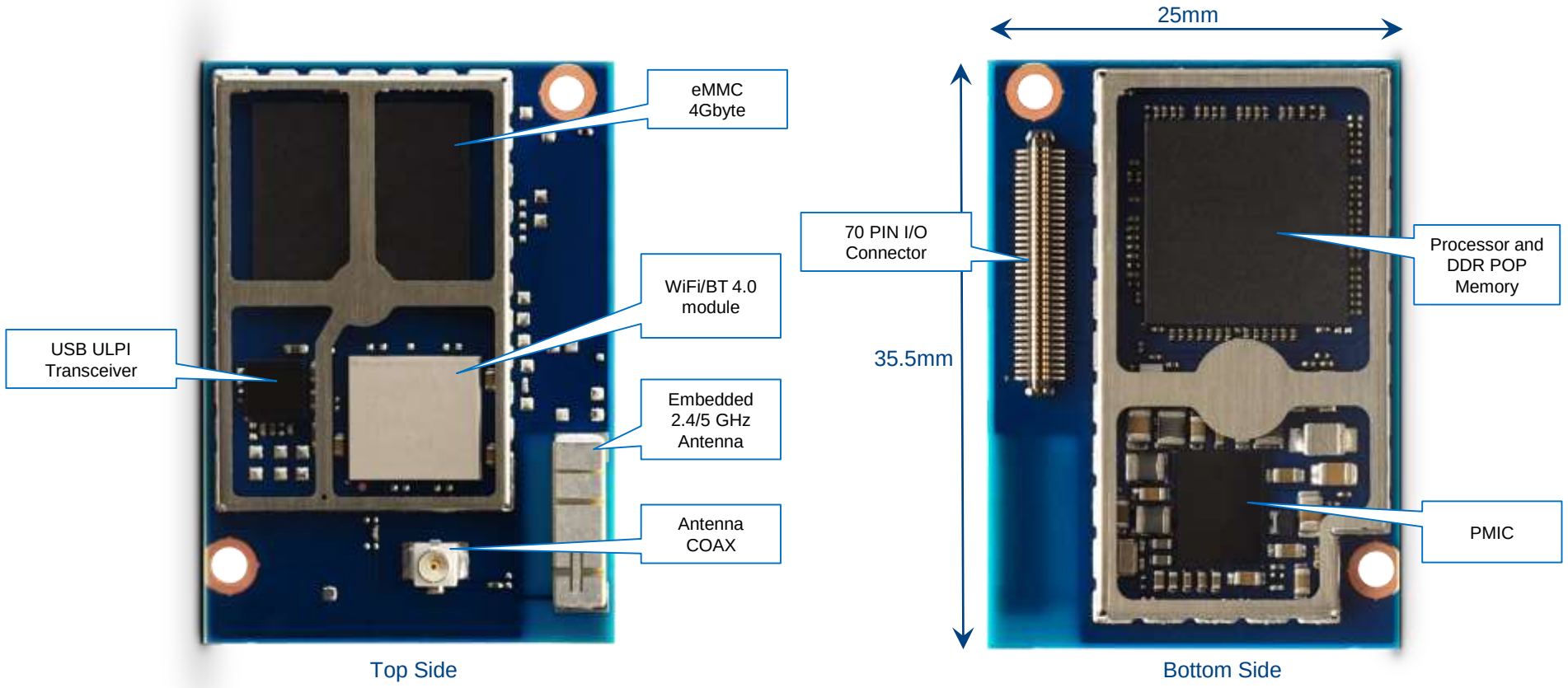
Agenda

- Intel Edison Introduction
- Arduino IDE
- Block Diagram of Pass Finder
- Making PWM and GPIO
- Connect WiFi and Broad casting Webpage
- USB Webcam Broadcasting
- Read Sensor from I2C
- Read Analog value from ADC

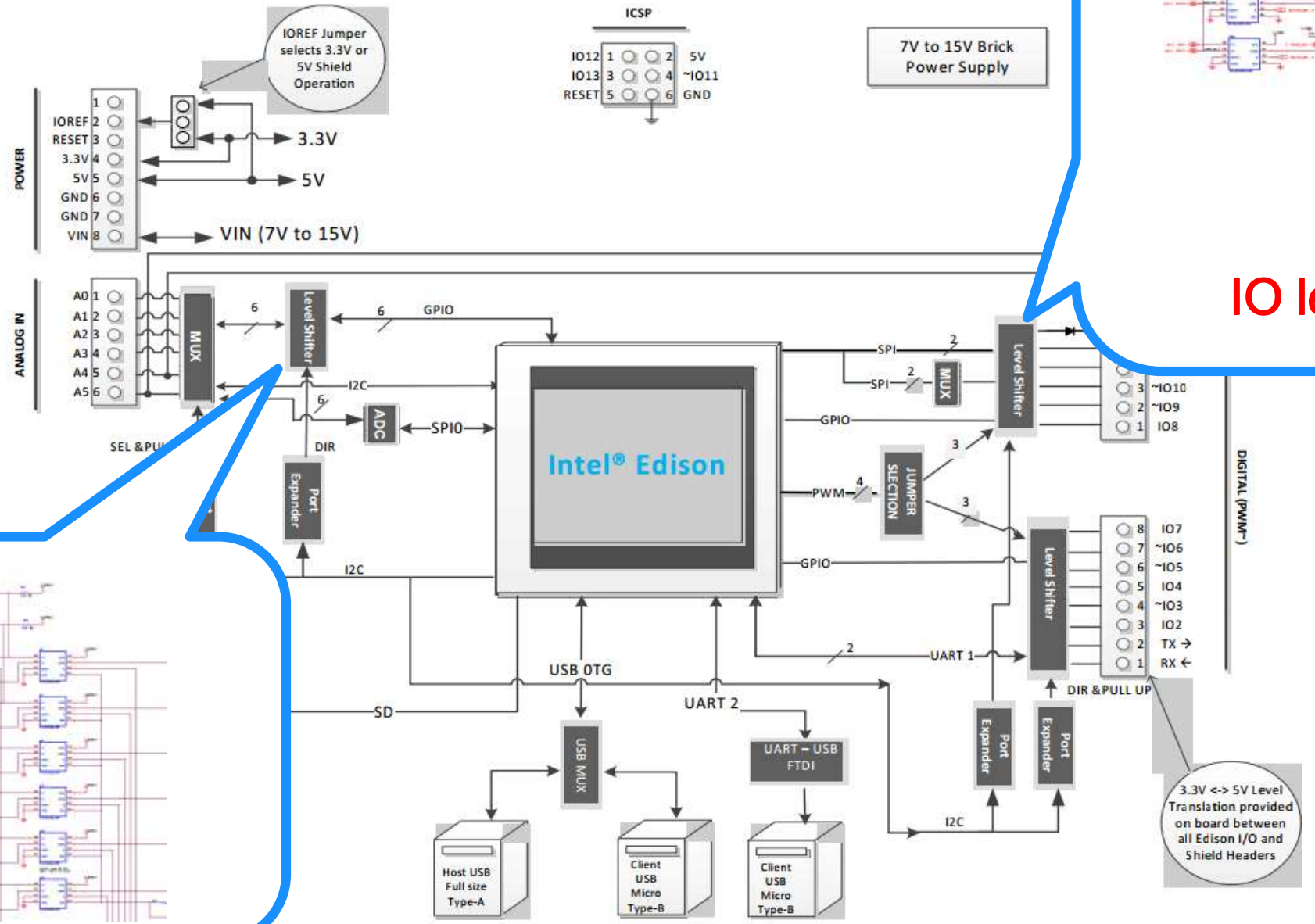
Intel Edison Introduction

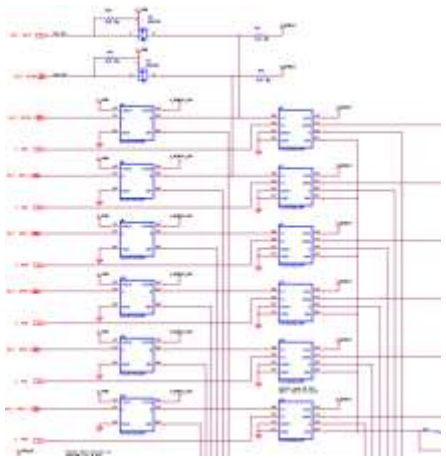


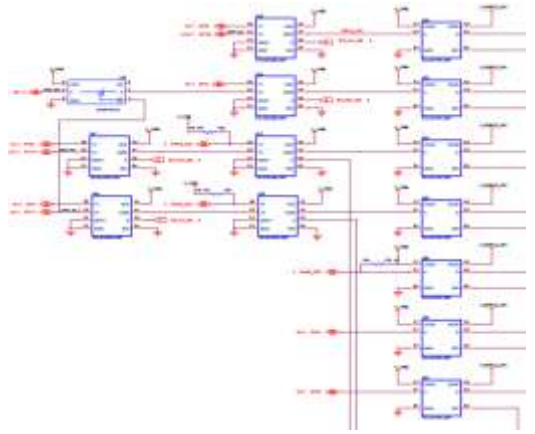
Intel Edison Introduction



Intel® Edison Kit for Arduino

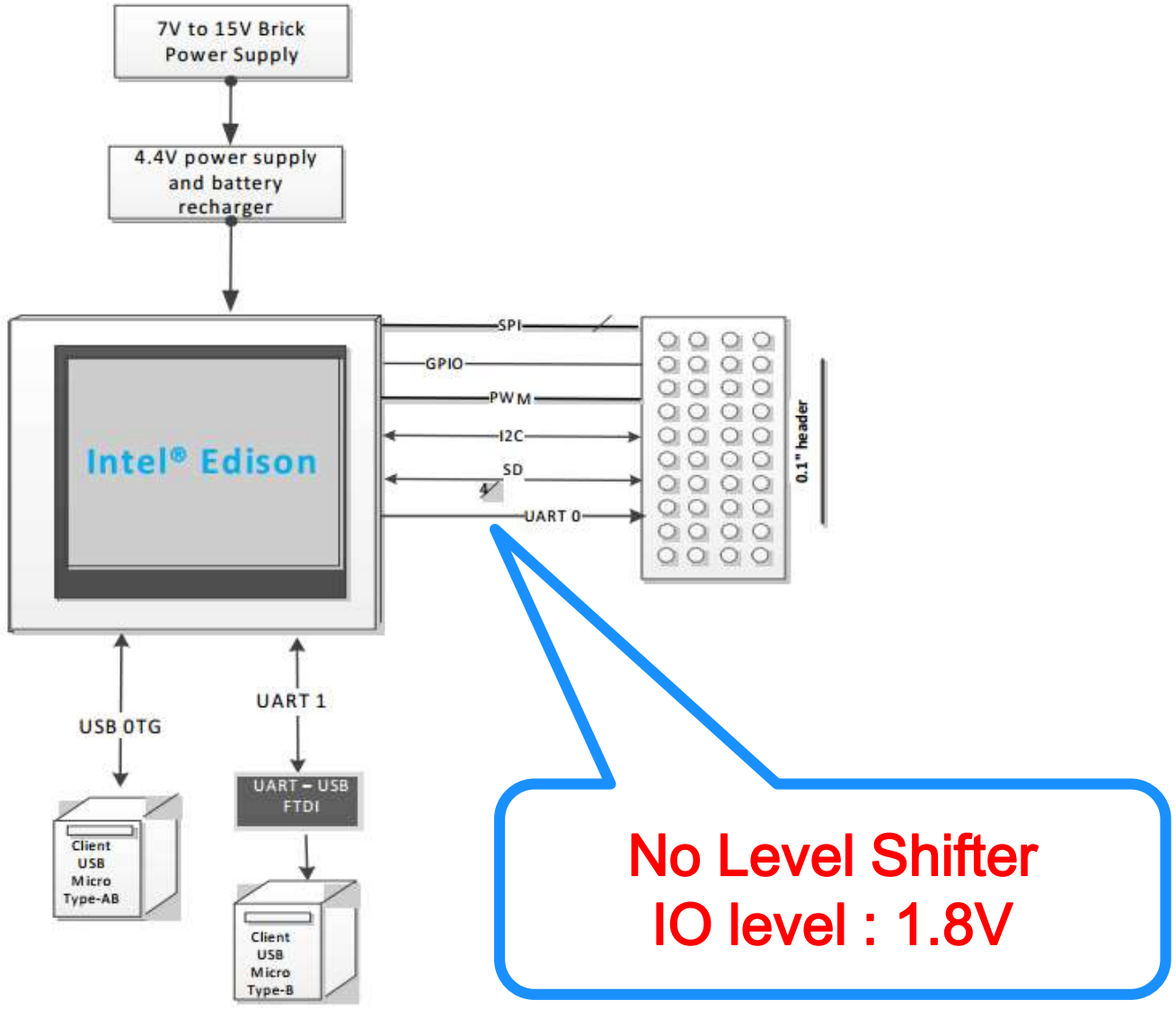






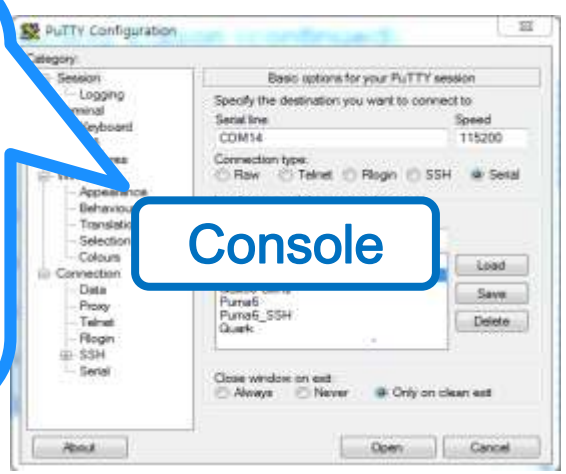
IO level : 5V

Intel® Edison Breakout Board Kit



Edison and PC Connection

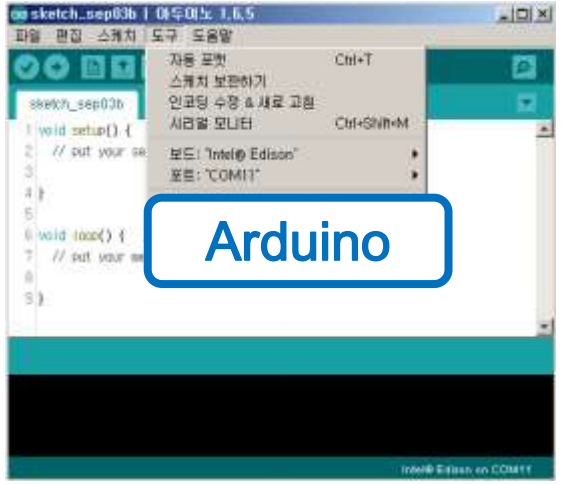
- Connection type: Serial
- Serial line: COM#
- USB Serial Port#
- Speed: 115200
- Session name: Edison
- Click on “Save” – “Open”.



Console

PC

USB



Arduino

Arduino

Console

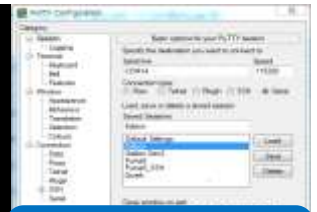
Edison and PC Connection – Basic Setup

```
Configure Edison: WiFi Connection

Scanning: 1 seconds leftt

0 :     Rescan for networks
1 :     Manually input a hidden SSID
2 :     \x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
3 :     UNIQUEST
4 :     UBCOM
5 :     UBCOM_5G
6 :     AndroidHotspot3787
7 :     AXIOS_MeetingRoom
8 :     AXIOS2
9 :     \x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
10 :    axios-guest
11 :    C1G2T2_UNICORN
12 :    giggle
13 :    AXIOS
14 :    explorer_5G
15 :    explorer_24
16 :    MAGNATEK


Enter 0 to rescan for networks.
Enter 1 to input a hidden network SSID.
Enter a number between 2 to 16 to choose one of the listed network SSIDs:
```

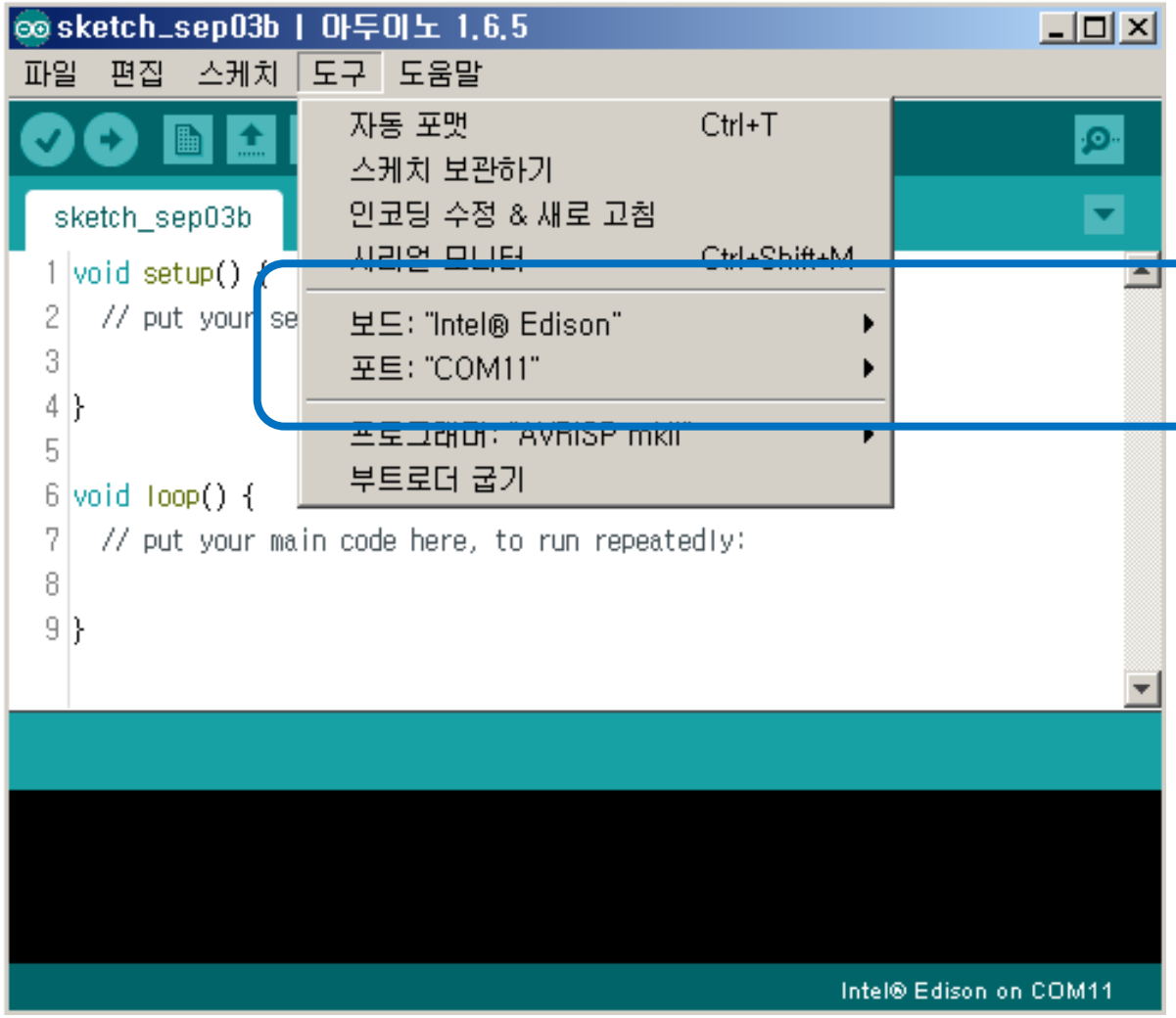


Console

- Connect two micro USB cables into the Arduino board.
- Open PuTTY and login
- edison# configure_edison --setup
- Type in a name for your Edison board.
- Generate root password.
- Type in “Y” and press Enter.
- Edison will scan for Wi-Fi networks for 10 seconds.
- Type in the number from the list for the network you’d like to connect to.
- Type in a password for the connection.
- Press Enter.

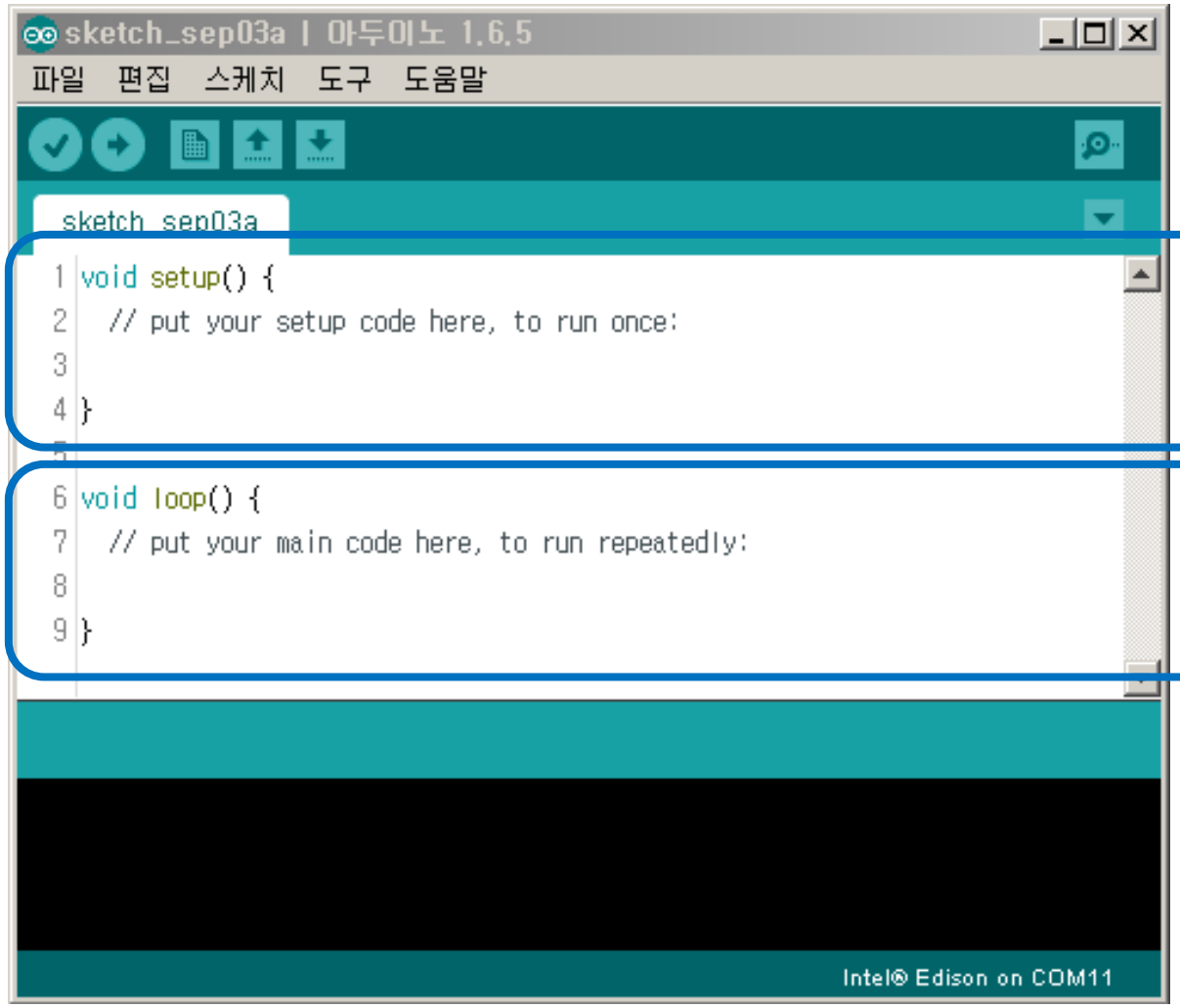
If need help, type “configure_edison --help“.

Arduino IDE



Board and Port matching

Arduino IDE



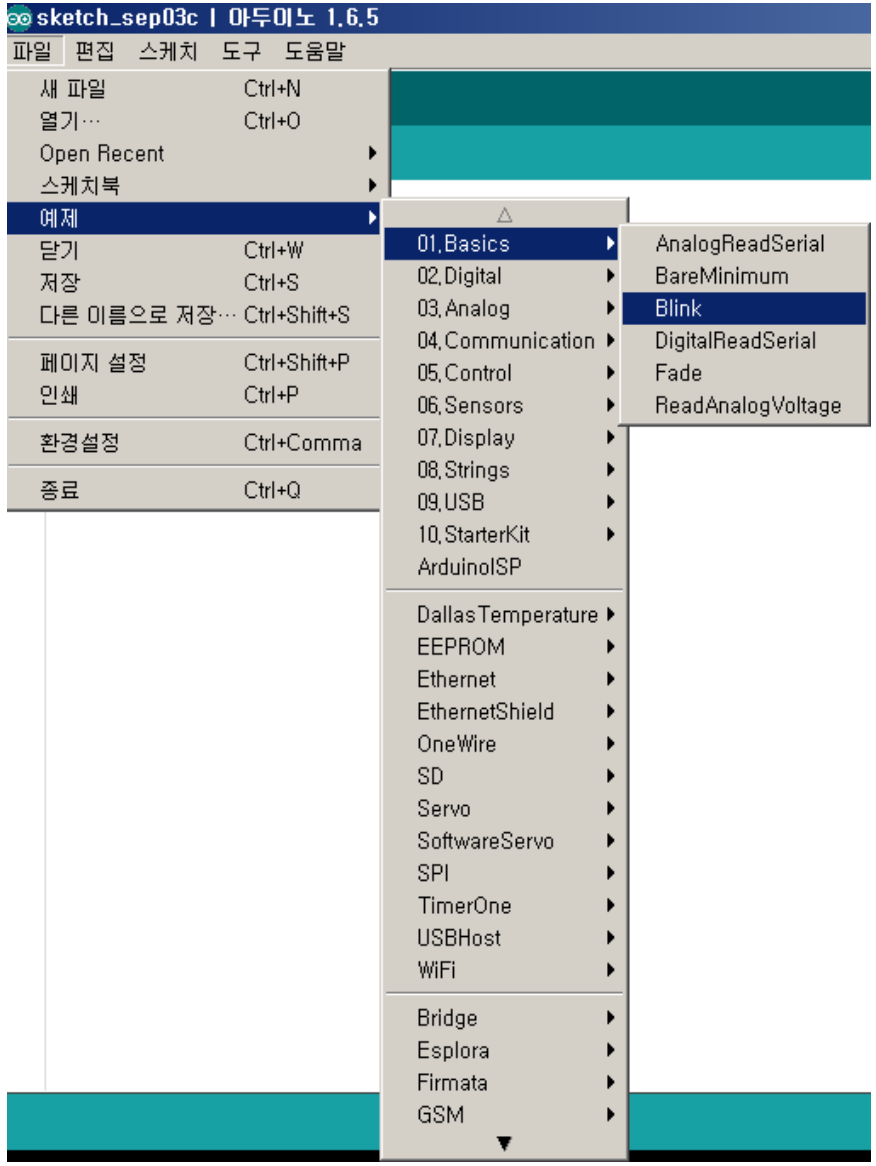
1 time Setup, 1 time Execution

Loop

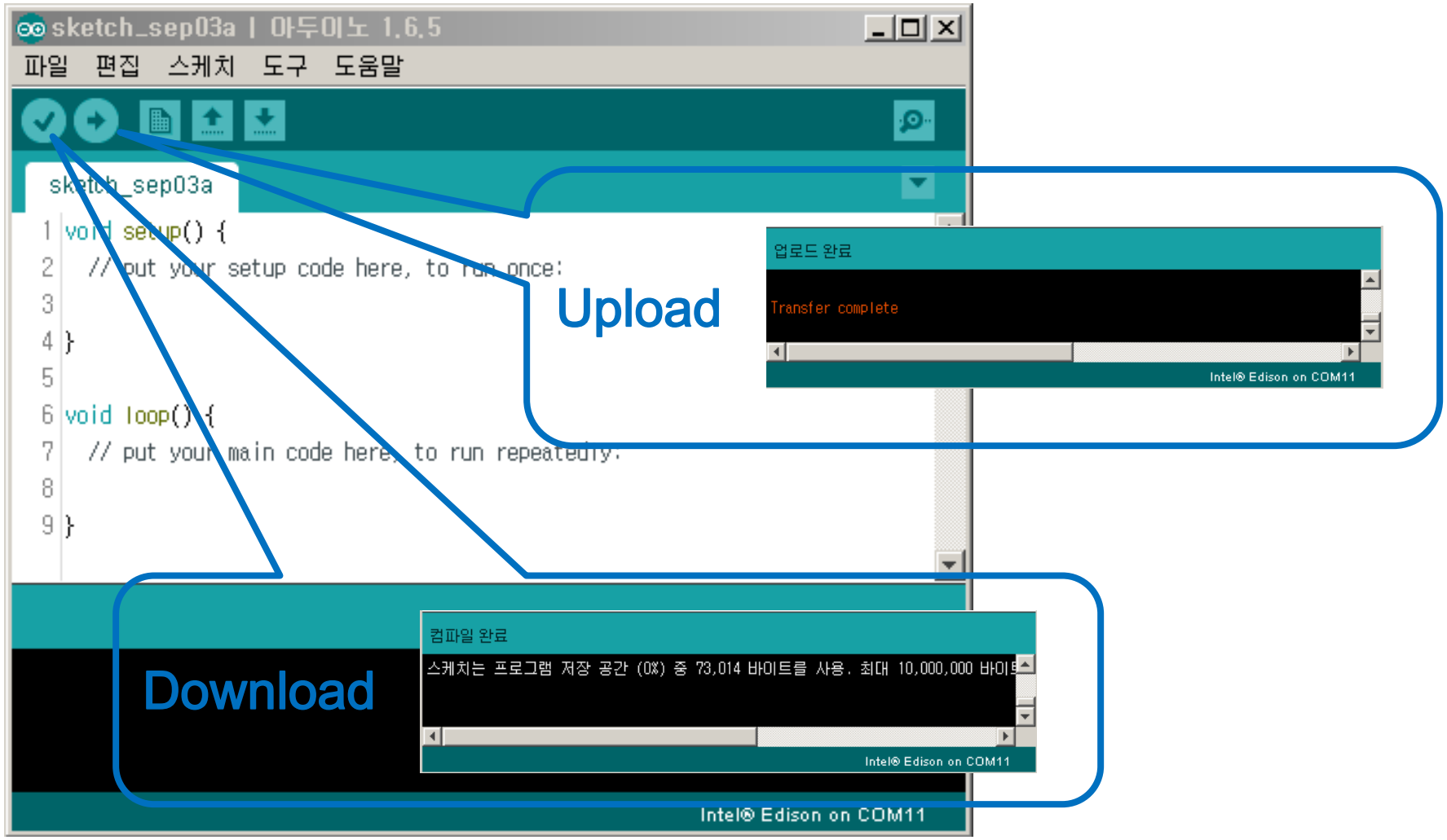
Arduino IDE

```
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);              // wait for a second
}
```

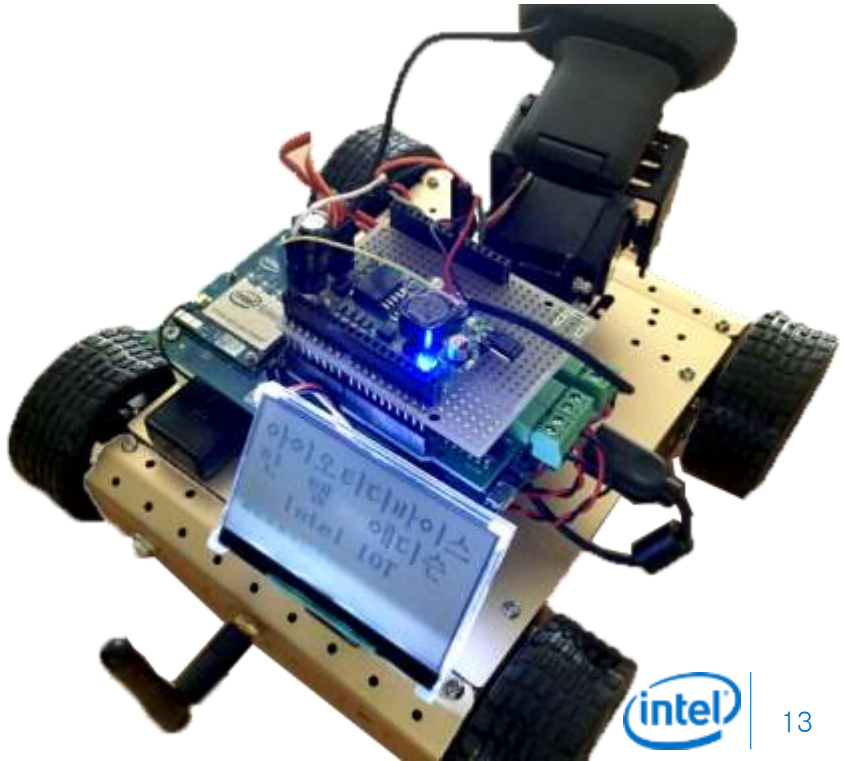


Arduino IDE

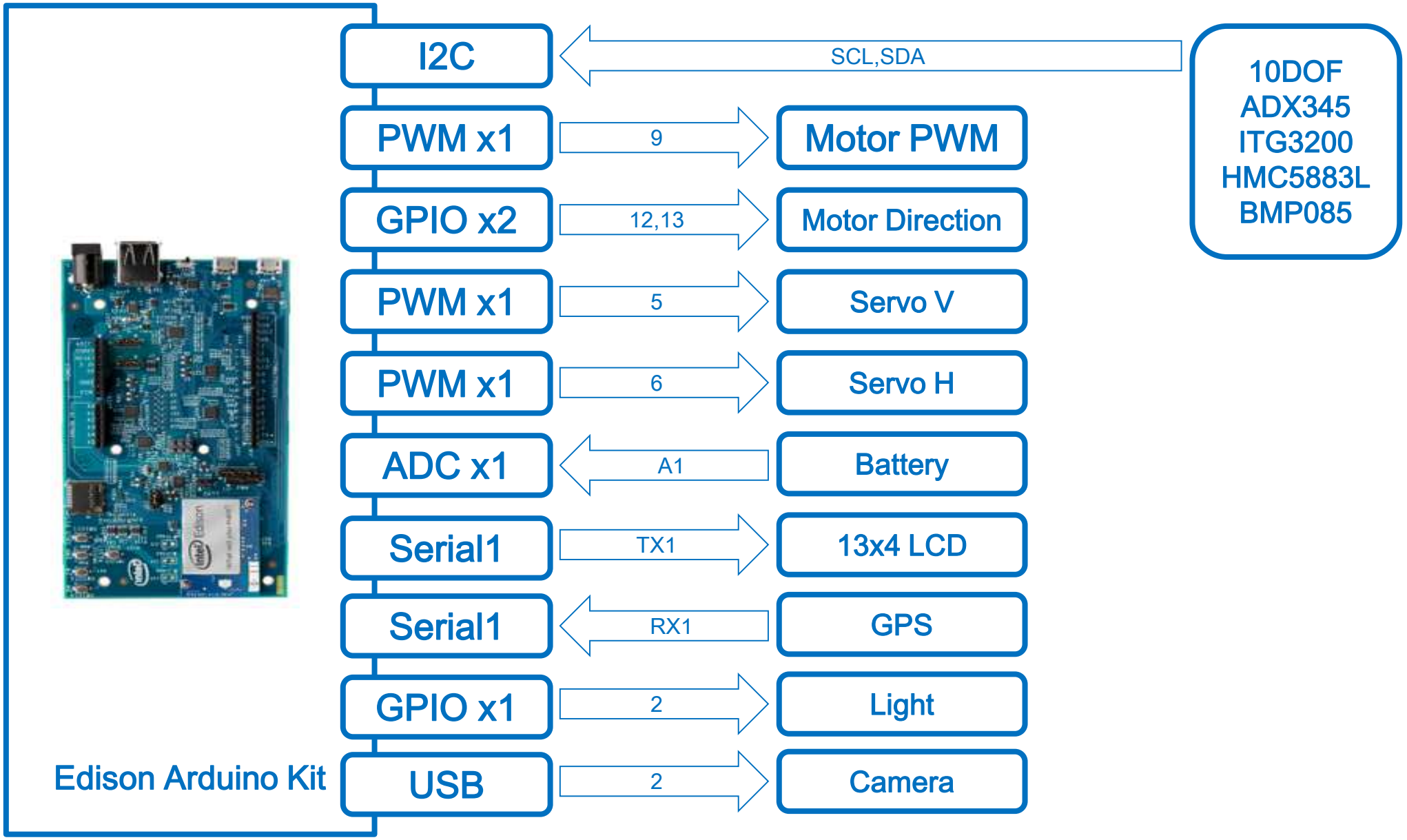

sketch_sep03a | 아두이노 1.6.5
파일 편집 스케치 도구 도움말
sketch_sep03a
1 void setup() {
2 // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7 // put your main code here, to run repeatedly:
8
9 }
Upload
업로드 완료
Transfer complete
Intel® Edison on COM11
Download
컴파일 완료
스케치는 프로그램 저장 공간 (0%) 중 73,014 바이트를 사용. 최대 10,000,000 바이
Intel® Edison on COM11
Intel® Edison on COM11

Pass Finder functions

- 2 Individual motor control with PWM and GPIO
- USB camera Streaming
- WiFi Connection
- Broadcasting Webpage
- Main battery monitoring with ADC
- Position reading from I2C Sensor
- Simple Serial LCD Control



Pass Finder Block Diagram



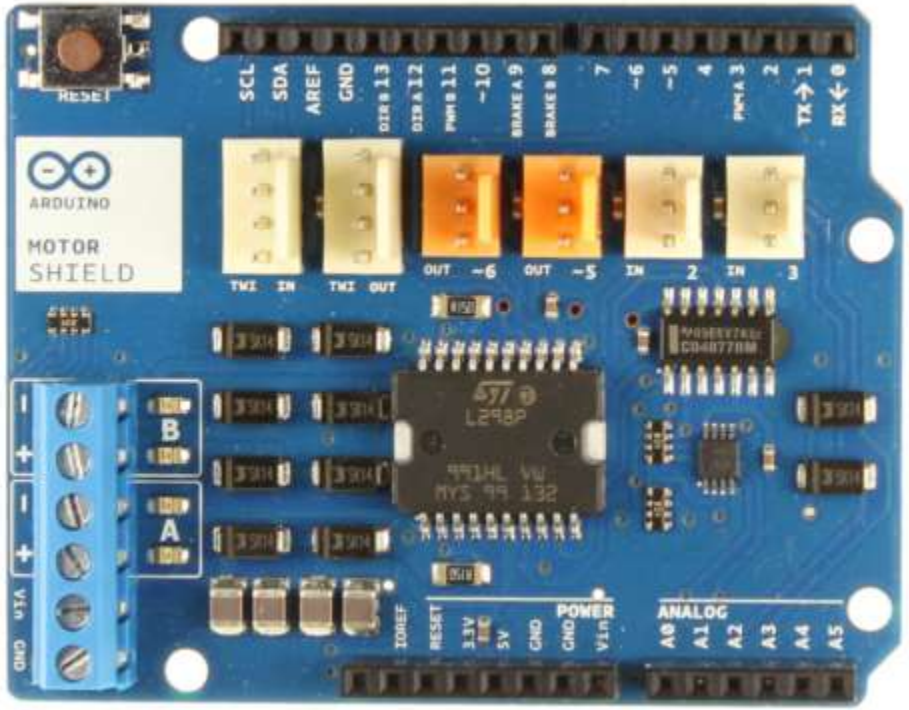
2 Individual motor control with PWM and GPIO

```
#define DIRA 12    // Motor A Direction
#define DIRB 13    // Motor B Direction

#define PWMA 3     // Motor A PWM
#define PWMB 11    // Motor B PWM

#define BREAKA   9  // Motor A Break      HIGH: Break  LOW: Run
#define BREAKB   8  // Motor B Break      HIGH: Break  LOW: Run

void setup()
{
  pinMode (DIRA, OUTPUT);
  pinMode (DIRB, OUTPUT);
  pinMode (BREAKA, OUTPUT);
  pinMode (BREAKB, OUTPUT);
}
```



2 Individual motor control with PWM and GPIO

```
void loop() {
  digitalWrite(BREAKA, LOW);  digitalWrite(BREAKB, LOW);  // Motor RUN
  digitalWrite(DIRA, HIGH);  analogWrite(PWMA, 200);
  digitalWrite(DIRB, HIGH);  analogWrite(PWMB, 200);
  delay(800);
  digitalWrite(BREAKA, HIGH);  digitalWrite(BREAKB, HIGH);  // Motor STOP
  delay(400);

  digitalWrite(BREAKA, LOW);  digitalWrite(BREAKB, LOW);  // Motor RUN
  digitalWrite(DIRA, LOW);  analogWrite(PWMA, 200);
  digitalWrite(DIRB, LOW);  analogWrite(PWMB, 200);
  delay(800);
  digitalWrite(BREAKA, HIGH);  digitalWrite(BREAKB, HIGH);  // Motor DTOP
  delay(400);
}
```

USB camera Streaming

UVC-compatible webcam (http://www.ideasonboard.org/uvc)

12V PSU

Web cam

ffmpeg

Web Server

Web page

Edison

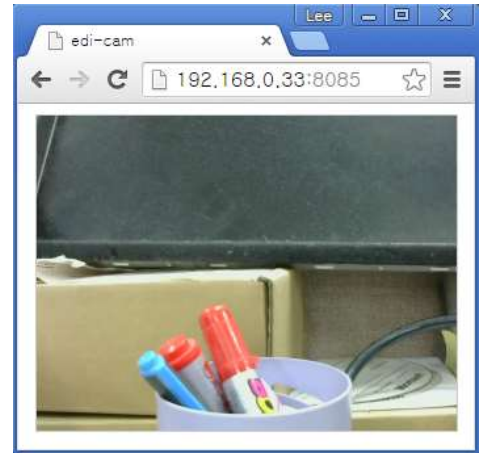
USB camera Streaming

- Edit base-feeds.conf
  - Vi /etc/opkg/base-feeds.conf
  - Press ‘i’ to edit
  - Enter
    - src/gz all http://repo.opkg.net/edison/repo/all
    - src/gz edison http://repo.opkg.net/edison/repo/edison
    - src/gz core2-32 http://repo.opkg.net/edison/repo/core2-32
  - Press ‘esc’ key
  - Press ‘:’
  - Press ‘wq’ and ‘enter’
- Update and install
  - Opkg update
  - Opkg install git
  - git clone https://github.com/drejkim/edi-cam.git
- Check and install optional
  - find /lib/modules/* -name 'uvc‘ → “/lib/modules/3.10.17-poky-edison+/kernel/drivers/media/usb/uvc”
  - No UVC module, install module with this command : opkg install kernel-module-uvcvideo

Do not install module with this result!!!

USB camera Streaming

- Check USB CAM attached to Edison
  - ls -l /dev/video0 → crw-rw---- 1 root video 81, 0 Dec 20 21:23 /dev/video0
- Install
  - Cd /etc/opkg/edi-cam/bin
  - ./install_ffmpeg.sh
  - cd /edi-cam/web/server
  - npm install
- Edit web page
  - vi /etc/opkg/edi-cam/web/client/index.htm
  - var wsUrl = 'ws://myedison.local:8084/'; → edit myedison to your Edison device name
- Run Server
  - Node /etc/opkg/edi-cam/web/server/server.js
- Connect web server
  - Your web server page is http://myedison.local:8080 or http://192.168.0.xx.local:8080 → Edison's IP address



WiFi Connection

- Connect WiFi from Arduino
  - Use basic Example from Arduino IDE “WiFiWebServer”
    - char ssid[] = "yourNetwork"; // your network SSID (name)
    - char pass[] = "secretPassword"; // your network password
    - int keyIndex = 0; // your network key Index number (needed only for WEP)
    - int status = WL_IDLE_STATUS;
    - WiFiServer server(80); → Fix to port 8080 to avoid conflict
  - How to disable Edison’s default web page → local address:80
    - vi /usr/lib/edison_config_tools/edison-config-server.js
    - Make skip this line “//http.createServer(requestHandler).listen(80);”

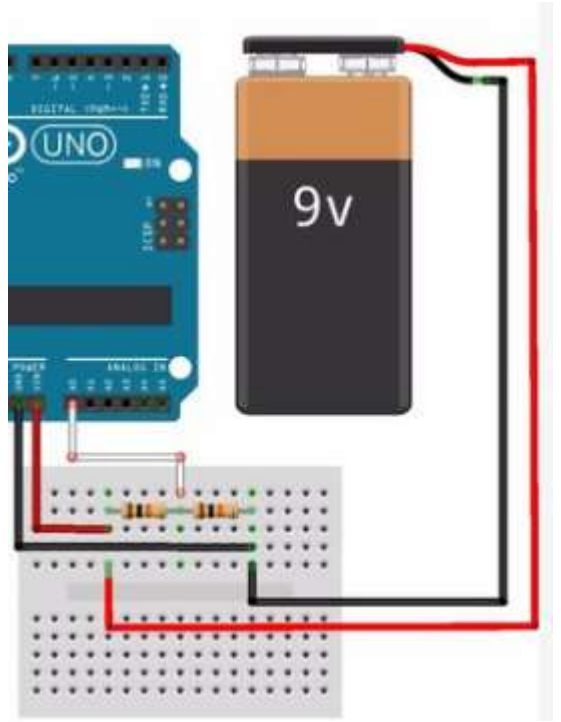
Broadcasting Webpage

```
// the content of the HTTP response follows the header:
          client.print("Click <a href=\"/H\">here</a> turn the LED on pin 9 on<br>");
          client.print("Click <a href=\"/L\">here</a> turn the LED on pin 9 off<br>");
          // The HTTP response ends with another blank line:
          client.println();
          // break out of the while loop:
          break;
        }
        else {      // if you got a newline, then clear currentLine:
          currentLine = "";}
      }
      else if (c != '\r') {    // if you got anything else but a carriage return character,
        currentLine += c;      // add it to the end of the currentLine
      }
      // Check to see if the client request was "GET /H" or "GET /L":
      if (currentLine.endsWith("GET /H")) {
        digitalWrite(9, HIGH);               // GET /H turns the LED on
      }
```

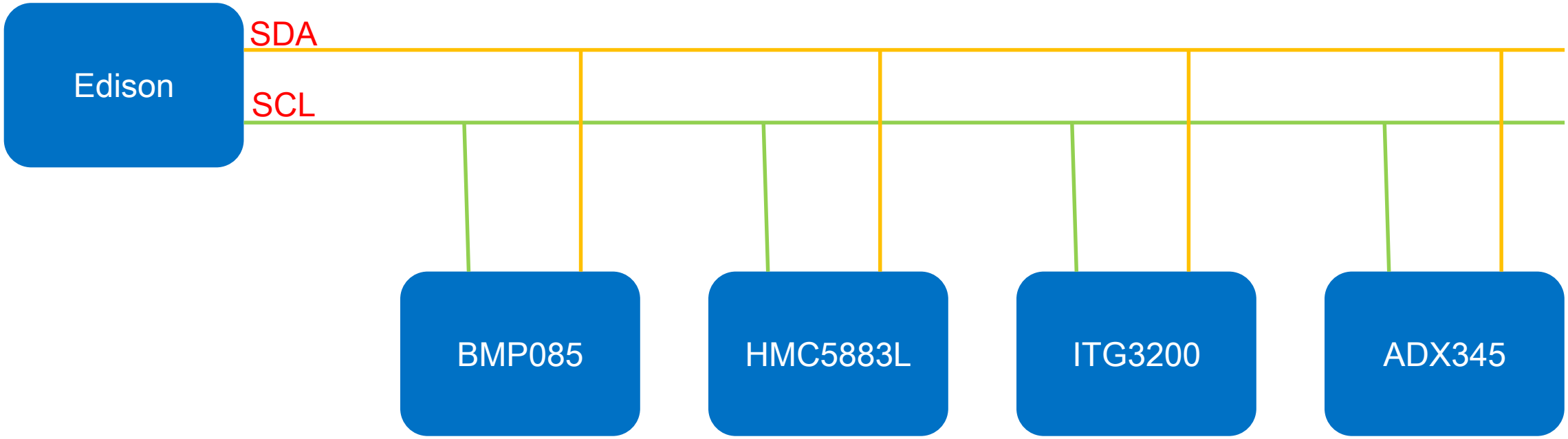
Main battery monitoring with ADC

```
// with Voltage Divider (2x 10K resistor)
void setup()
{
}
void loop()
{
  printVolts();
}

 void printVolts()
{
  int sensorValue = analogRead(A0); //read the A0 pin value
  float voltage = sensorValue * (5.00 / 1023.00) * 2; //convert the value to a true voltage.
  lcd.setCursor(0,0);
  lcd.print("voltage = ");
  lcd.print(voltage); //print the voltage to LCD
  lcd.print(" V");
}
```

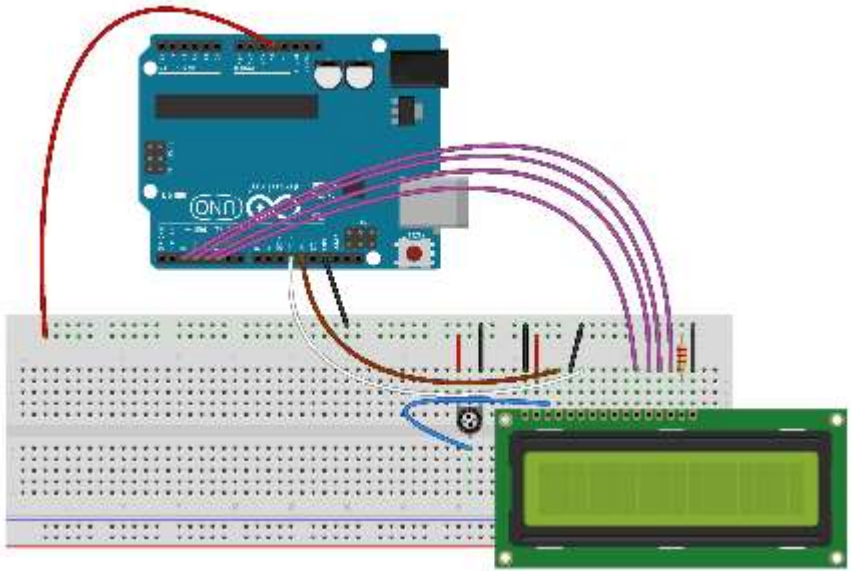


Position reading from I2C Sensor



Simple Serial LCD Control

- Serial type or data type?
  - #include <LiquidCrystal.h>
  - // initialize the library with the numbers of the interface pins
  - LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
  - void setup() {
  - // set up the LCD's number of columns and rows:
  - lcd.begin(16, 2);
  - // Print a message to the LCD.
  - lcd.print("hello, world!");
  - }
  - void loop() {
  - // set the cursor to column 0, line 1
  - // (note: line 1 is the second row, since counting begins with 0):
  - lcd.setCursor(0, 1);
  - // print the number of seconds since reset:
  - lcd.print(millis()/1000);
  - }



Reference

Intel Edison Download and documentation
http://www.intel.co.kr/content/www/kr/ko/do-it-yourself/downloads-and-documentation.html

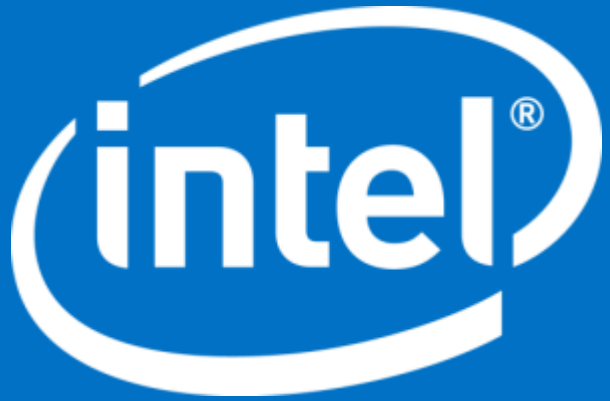
Intel Edison Project Gallery
https://communities.intel.com/community/makers/edison/project_gallery?_ga=1.180645666.484733669.1437738635

SparkFun learn
https://learn.sparkfun.com/

Getting Started:
https://communities.intel.com/community/makers/edison/getting-started

Software Downloads:
https://communities.intel.com/docs/DOC-23242

Forums:
https://communities.intel.com/community/makers/edison/forums



Appendix

Recovering Edison – Console

If Password Is Unknown

- Copy all of the latest images to Edison volume in computer.
- Open PuTTY.
- Reboot Edison leaving the serial connection opened, once you start to see the boot up message hit any key to stop autoboot.

```
microkernel built 23:15:13 Apr 24 2014

******* PSH loader *******
PCM page cache size = 192 KB
Cache Constraint = 0 Pages
Arming IPC driver ..
Adding page store pool ..
PagestoreAddr(IMR Start Address) = 0x04899000
pageStoreSize(IMR Size)          = 0x00080000

*** Ready to receive application ***


U-Boot 2014.04 (Sep 08 2014 - 14:09:10)

       Watchdog enabled
DRAM:  980.6 MiB
MMC:   tangier_sdhci: 0
In:    serial
Out:   serial
Err:   serial
Hit any key to stop autoboot:  0
```

- Type in “do run_ota”

```
Hit any key to stop autoboot:  0
boot > run do_ota
```

- After finishing upgrading, you need to set up Wi-Fi configuration again.

GPIO Pin Multiplexing Guide

Shield Pin	GPIO Linux Pin	PWM Linux Pin	Muxed functions	Notes
IO0	130		UART1_RXD	
IO1	131		UART1_TXD	
IO2	128		UART1_CTS*	
IO3	12	0	PWM0	Depends on PWM Swizzler**
IO4	129		UART1_RTS*	
IO5	13	1	PWM1	Depends on PWM Swizzler**
IO6	182	2	PWM2	Depends on PWM Swizzler**
IO7	48		-	
IO8	49		-	
IO9	183	3	PWM3	Depends on PWM Swizzler**
IO10	41	Swiz	SPI_2_SS1	
			I2S_2_FS*	
			PWM4_OUT	Depends on PWM Swizzler**

GPIO Pin Multiplexing Guide

IO11	43	Swiz	SPI_2_TXD	
			I2S_2_TXD*	
			PWM5_OUT	Depends on PWM Swizzler**
IO12	42		SPI_2_RXD	
			I2S_2_RXD*	
IO13	40		SPI_2_CLK	
			I2S_2_CLK*	
IO14	44		AIN0	
IO15	45		AIN1	
IO16	46		AIN2	
IO17	47		AIN3	
IO18	14		AIN4	
			I2C_6_SDA	
IO19	165		AIN5	

GPIO Pin Multiplexing Guide

	Linux GPIO Pin	GPIO Pin Mux			SoC Pin Modes		Output Enable * (high = output)	Pull-up Enable**
		Linux Pin	0 (low)	1 (high)	0	1	Linux	Linux
IO0	130				GPIO	UART	248	216
IO1	131				GPIO	UART	249	217
IO2	128				GPIO	UART	250	218
IO3	12				GPIO	PWM	251	219
IO4	129				GPIO	UART	252	220
IO5	13				GPIO	PWM	253	221
IO6	182				GPIO	PWM	254	222
IO7	48				GPIO		255	223
IO8	49				GPIO		256	224
IO9	183				GPIO	PWM	257	225
IO10	41	263	PWM	see 240	GPIO	I2S or SPI	258	226
		240	GPIO or I2S	GPIO or SPI_FS				

GPIO Pin Multiplexing Guide

IO11	43	262	PWM	see 241	GPIO	I2S or SPI	259	227
		241	GPIO or I2S	GPIO or SPI TXD				
IO12	42	242	GPIO or I2S	GPIO or SPI RXD	GPIO	I2S or SPI	260	228
IO13	40	243	GPIO or I2S	GPIO or SPI CLK	GPIO	I2S or SPI	261	229
IO14 (A0)	44	200	GPIO	A0	GPIO		232	208
IO15 (A1)	45	201	GPIO	A1	GPIO		233	209
IO16 (A2)	46	202	GPIO	A2	GPIO		234	210
IO17 (A3)	47	203	GPIO	A3	GPIO		235	211
IO18 (A4)	14	204	GPIO or I2C SDA	A4	GPIO	I2C-6	236	212
IO19 (A5)	165	205	GPIO or I2C SCL	A5	GPIO	I2C-6	237	213

GPIO Pin Multiplexing Guide

- **GPIO allocation and shield pin control**
  - Identify the Arduino shield pin number of the pin you wish to use, in the range IO0 – IO19.
  - Identify the functions available for the given pin, and select the function you wish to use.
    - ✓ Typical functions are GPIO, PWM, UART, I2C, SPI, ADC.
    - ✓ Only some functions are available on each pin.
  - Determine which GPIO signals, if any, need to be configured to select the correct pin muxing option for the selected function. Some pins only have a single function, or do not require mux control.
  - Determine which GPIO signals, if any, need to be configured to select the buffer direction for input or output, and determine the direction that is required.
  - Determine which GPIO signals, if any, need to be configured to select the pull-up resistor control, and whether the pull-up resistor should be enabled or disabled. Generally, for most pin functions, the pull-up resistors should typically be disabled. For GPIO input functions, the pull-up resistor may optionally be enabled or disabled according to the needs of the user.
  - Export the above GPIO numbers for access in the Linux user-space environment (i.e. from the command shell).
  - Configure the above GPIO numbers for output.
  - Assert the TRI_STATE_ALL signal to disconnect the shield pins.
  - Set the above GPIO numbers to assert their output logic levels as high or low.
  - Set the SoC GPIO pin mode for the required functionality.
  - De-assert the TRI_STATE_ALL signal to reconnect the shield pins

GPIO Pin Multiplexing Guide

- **Configuring IO6 as a PWM output**
  - The shield number is IO6 and the GPIO number is 182.
  - The function required is PWM. Other function on the pins is GPIO.
  - SoC pin mode must be set to ‘mode1’ to select PWM.
  - GPIO254 must be set to 1 to enable the output direction for IO6.
  - GPIO222 must be set as a high-impedance input to disable the external pull-up resistor for IO6.
  - The TRI_STATE_ALL signal is controlled by GPIO 214.
    - ✓ edison# echo 254 > /sys/class/gpio/export
    - ✓ edison# echo 222 > /sys/class/gpio/export
    - ✓ edison# echo 214 > /sys/class/gpio/export
    - ✓ edison# echo low > /sys/class/gpio/gpio214/direction
    - ✓ edison# echo high > /sys/class/gpio/gpio254/direction
    - ✓ edison# echo in > /sys/class/gpio/gpio222/direction
    - ✓ edison# echo mode1 > /sys/kernel/debug/gpio_debug/gpio182/current_pinmux
    - ✓ edicon# echo high > /sys/class/gpio/gpio214/direction

GPIO Pin Multiplexing Guide

- **Configuring IO6 as a PWM output**
  - Now, it should be possible to use IO6 as a PWM output. For example:
    - ✓ edison# echo 2 > /sys/class/pwm/pwmchip0/export
    - ✓ edison# echo 2000000 > /sys/class/pwm/pwmchip0/pwm2/duty_cycle
    - ✓ edison# echo 1 > /sys/class/pwm/pwmchip0/pwm2/enable