EPICS on Raspberry-Pi

2017/5/15

M. Iwasaki (Osaka City University)

The original title of this hands-on was “Install EPICS on Raspberry Pi”.

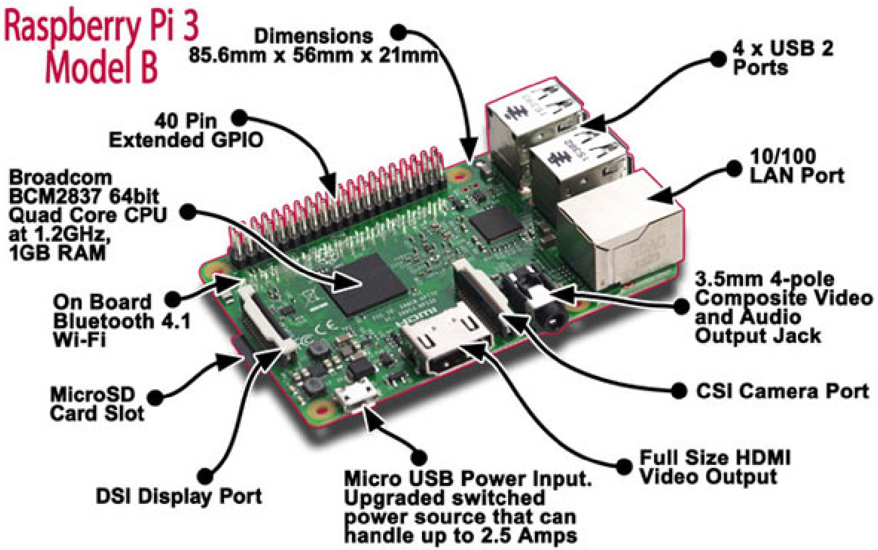
But we only have 1-1.5 hour for this hands on ,
and it is impossible to install the EPICS within this session.
(It takes ~1 hour to compile the EPICS etc on RPi.)

→ We use the EPICS-installed RPi for the hands-on.
Here you’ll
1) SSH from your note PC to RPi
2) Run the IOC shell on RPi to control the Arduino
3) caget / put / monitor from your note PC

Raspberry Pi

The Raspberry Pi (RPi) is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation

In this hands on, we use RPi3 model B (2016 Feb.)



Raspberry Pi 3 Model B

Dimensions 85.6mm x 56mm x 21mm

4 x USB 2 Ports

40 Pin Extended GPIO

10/100 LAN Port

Broadcom BCM2837 64bit Quad Core CPU at 1.2GHz, 1GB RAM

3.5mm 4-pole Composite Video and Audio Output Jack

On Board Bluetooth 4.1 Wi-Fi

CSI Camera Port

MicroSD Card Slot

Full Size HDMI Video Output

DSI Display Port

Micro USB Power Input. Upgraded switched power source that can handle up to 2.5 Amps

Raspberry Pi

The Raspberry Pi (RPi) is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation

In this hands on, we use RPi3 model B (2016 Feb.)

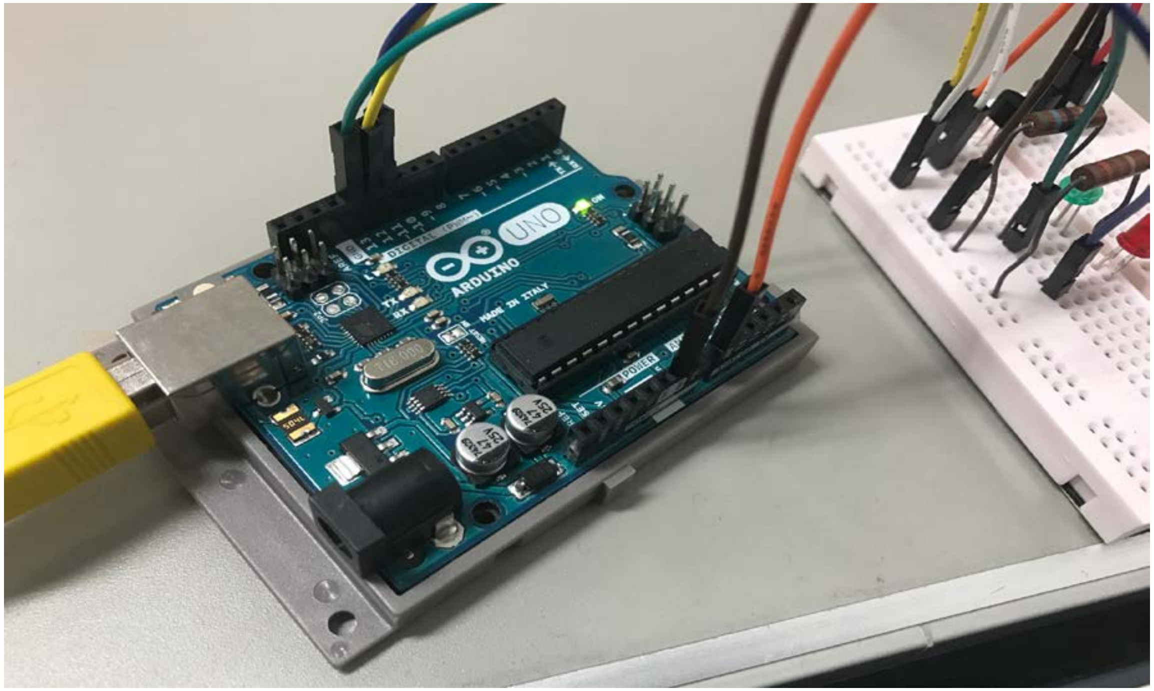
- A 1.2GHz 64-bit quad-core ARMv8 CPU
- 1GB RAM
- 802.11n Wireless LAN
- Ethernet port
- **4 USB ports**
- HDMI port
- **40 GPIO pins**, etc,,

Rpi3 controls the device via Ethernet, USB, or GPIO.

Today we use RPi as an IOC
to control the Arduino via USB.
Tomorrow, there is a talk on GPIO

Arduino

Arduino boards consist of an ATmel 8-bit AVR microcontroller with varying amounts of flash memory, pins, and features.

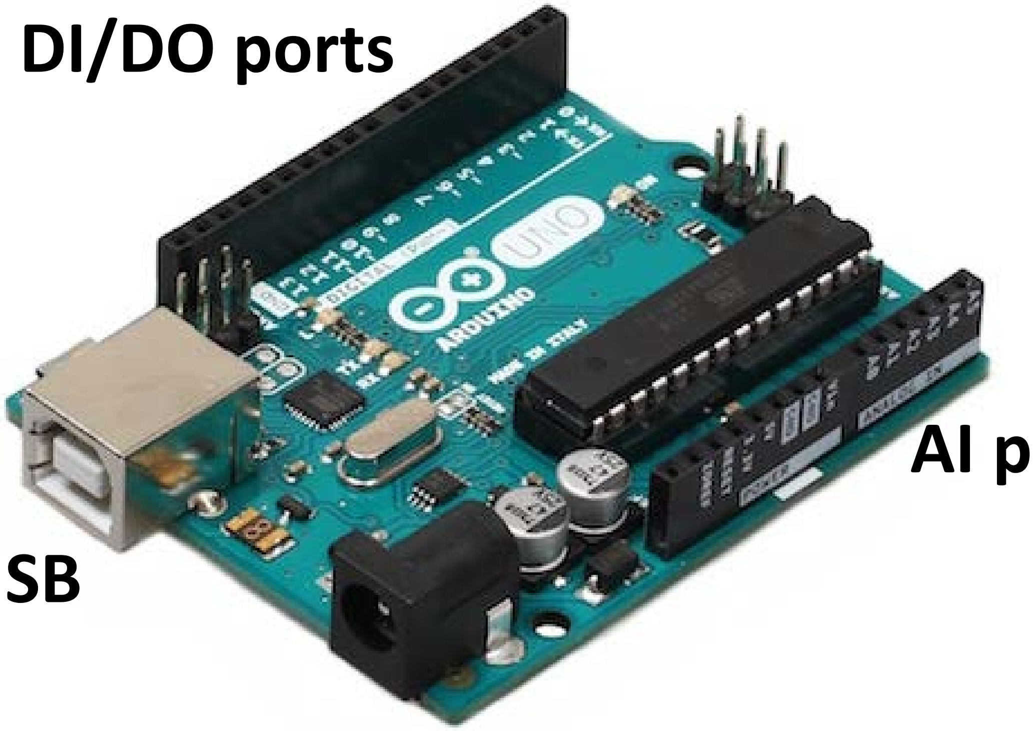


Arduino Uno

- ATmega328P.
- 14 Digital I/O pins
- 6 Analog inputs
- A 16 MHz quartz crystal
- 1 USB connection

Arduino

Arduino boards consist of an ATmel 8-bit AVR microcontroller with varying amounts of flash memory, pins, and features.

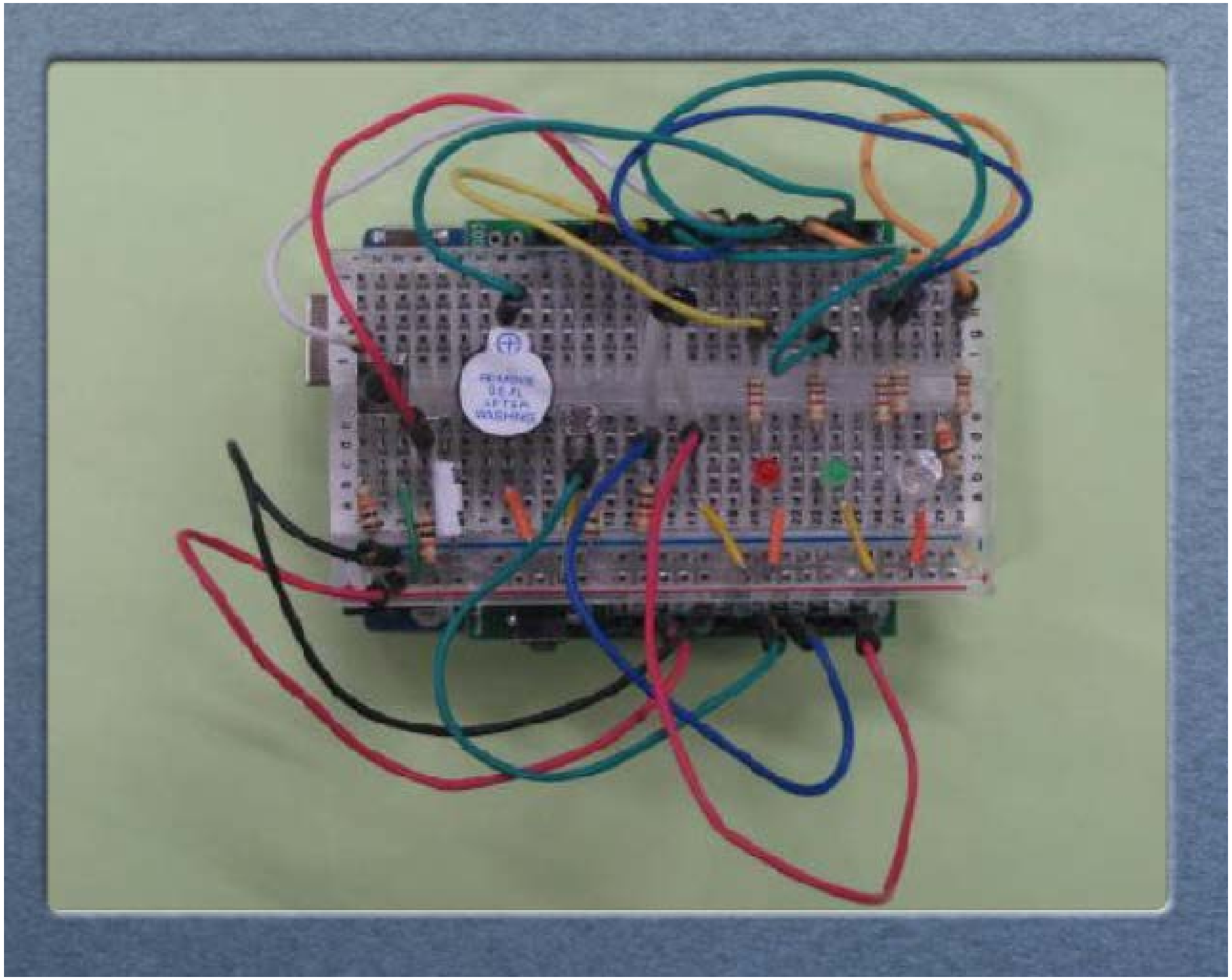


Arduino Uno

- ATmega328P.
- 14 Digital I/O pins
- 6 Analog inputs
- A 16 MHz quartz crystal
- 1 USB connection

Arduino shield(ext. board) with Bread board for EPICS training

N. Yamamoto (KEK/J-PARC)



Includes

1 TriColor LED (RGB)

2 LEDs (Red, Green)

1 Buzzer

1 Push Switch

1 Thermistor

1 CdS (Luminance sensor)

1 level Switch

~5,000 JPY

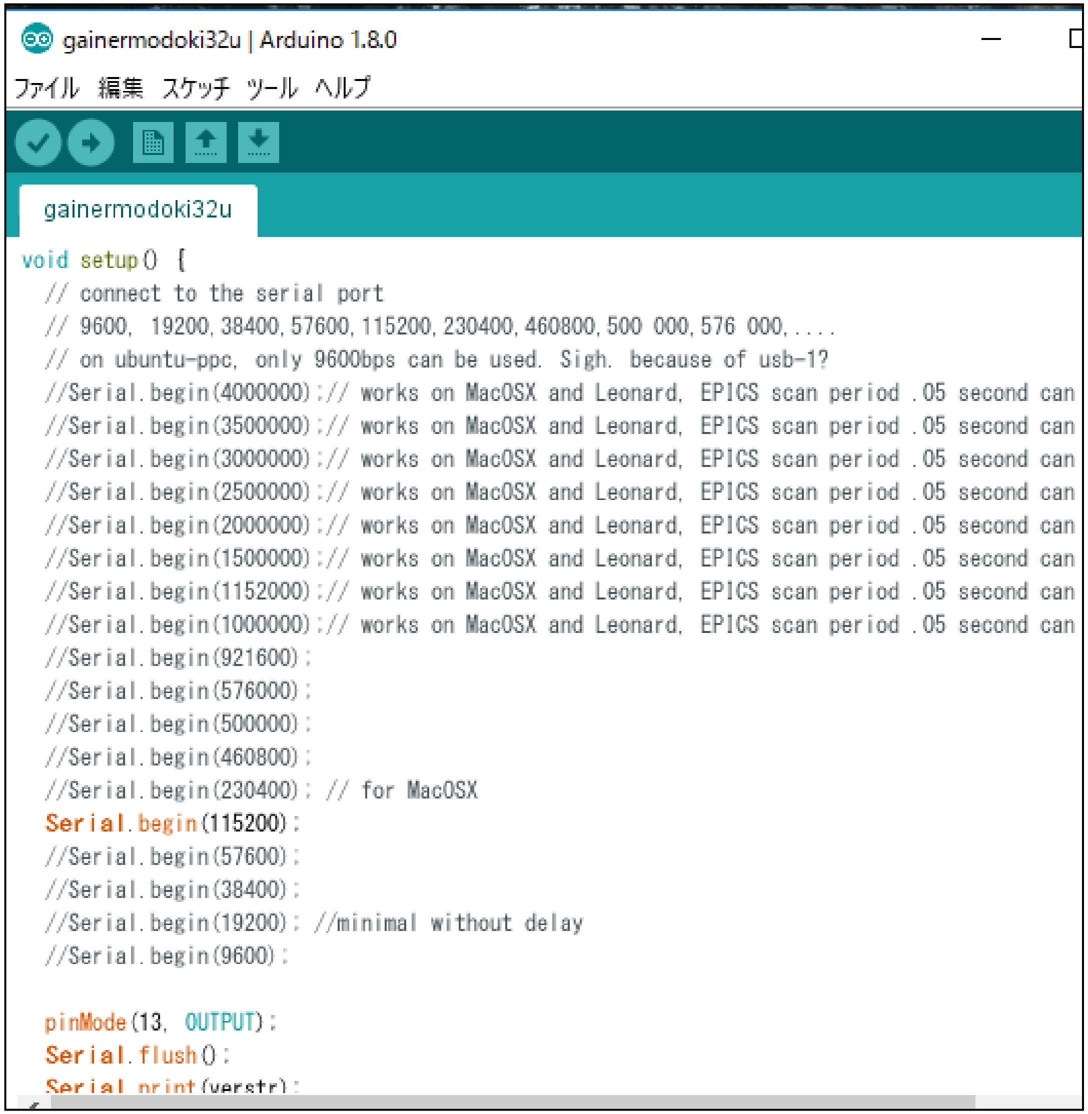
Including Arduino

Arduino sketch file

We use **Arduino IDE**
to write code (=sketch file), and
upload it to the board

- Open-source Arduino Software
- Runs on Windows, Mac OS X, Linux

This time we use
Gainermodoki.ino
= Sketch file for the Arduino shield
(in previous page) for the hands-on
by N.Yamamoto.


gainermodoki32u | Arduino 1.8.0
ファイル 編集 スケッチ ツール ヘルプ
gainermodoki32u

```
void setup() {
  // connect to the serial port
  // 9600,  19200, 38400, 57600, 115200, 230400, 460800, 500 000, 576 000, . . . .
  // on ubuntu-ppc,  only 9600bps can be used.  Sigh.  because of usb-1?
  //Serial.begin(4000000);// works on MacOSX and Leonard,  EPICS scan period .05 second can
  //Serial.begin(3500000);// works on MacOSX and Leonard,  EPICS scan period .05 second can
  //Serial.begin(3000000);// works on MacOSX and Leonard,  EPICS scan period .05 second can
  //Serial.begin(2500000);// works on MacOSX and Leonard,  EPICS scan period .05 second can
  //Serial.begin(2000000);// works on MacOSX and Leonard,  EPICS scan period .05 second can
  //Serial.begin(1500000);// works on MacOSX and Leonard,  EPICS scan period .05 second can
  //Serial.begin(1152000);// works on MacOSX and Leonard,  EPICS scan period .05 second can
  //Serial.begin(1000000);// works on MacOSX and Leonard,  EPICS scan period .05 second can
  //Serial.begin(921600);
  //Serial.begin(576000);
  //Serial.begin(500000);
  //Serial.begin(460800);
  //Serial.begin(230400);  // for MacOSX
  Serial.begin(115200);
  //Serial.begin(57600);
  //Serial.begin(38400);
  //Serial.begin(19200);  //minimal without delay
  //Serial.begin(9600);

  pinMode(13,  OUTPUT);
  Serial.flush();
  Serial.print(verstr);
```

Arduino sketch file
/opt/epics/modules/soft/StreamDevice/apps/protocols/gainermodoki/gainermodoki32u/
gainermodoki32u.ino

RPi Set Up

What we already did for this hands-on :

1. **Install OS (raspbian) into RPi**

   Download the image of 2017-04-10-raspbian-jessie
   Copy to the microSD card

2. **Install EPICS into RPi**

   EPICS base, asyn, stream device, etc. under /opt/epics
   This time, we install the files compiled by N.Yamamoto.

3. **Arduino setup**

   Install and setup by the command
   “sudo apt-get install arduino”

RPi Hands On

1. Divide into 5 (or 6) groups

2. SSH to RPi and run the IOC shell

In each group, only 1 person can edit the file and run the IOC shell.

- SSH to RPi (We'll tell the user/ password / IP address later.)
- cd ~/For_HandsOn/scripts
- For group 2-6, edit the ArduinoProto.cmd to change the record name.

```
From   P=myArduuno1,,,
To     P=myArduuno*,,,    (*=group number, 1-6)
```

← nano ArduinoProto.cmd

- Run the IOC shell

```
./ArduinoProto.cmd
```

(Directory is linked from/opt/epics/modules/soft/StreamDevice/apps/)

RPi Hands On

3. Access the EPICS records from your note PC

- caget / camonitor / caput <PV name>
- Make the MEDM window to monitor / access the PVs

myArduino*:LUM	Analog Val. From LUM sensor	
myArduino*:LED:R	Red LED	(=0 Turn On =1 Turn Off)
myArduino*:LED:G	Green LED	(=0 Turn On =1 Turn Off)
myArduino*:CLED:R	3-Col. LED Red	(=0 Turn On =1 Turn Off)
myArduino*:CLED:G	3-Col. LED Green	(=0 Turn On =1 Turn Off)
myArduino*:CLED:B	3-Col. LED Blue	(=0 Turn On =1 Turn Off)
myArduino*:buzzer	(1: beep sound on)	

(* = group number, 1-6)