Smart Software Project

Lab: Week 9
Ultrasonic Sensors

Prof. HyungJune Lee
hyungjune.lee@ewha.ac.kr

이화여자대학교
EWHA WOMANS UNIVERSITY

Today

- Lab announcement
- Ultrasonic sensor
- Lab assignment #6
- Course announcement

Class Schedule

Week	Lecture Contents	Lab Contents
Week 1	Course introduction	Arduino introduction: platform & programming environment
Week 2	Embedded system overview & source management in collaborative repository (using GitHub)	Lab 1: Arduino Mega 2560 board & SmartCAR platform
Week 3	ATmega2560 Micro-controller (MCU): architecture & I/O ports, Analog vs. Digital, Pulse Width Modulation	Lab 2: SmartCAR LED control
Week 4	Analog vs. Digital & Pulse Width Modulation	Lab 3: SmartCAR motor control (Due: HW on creating project repository using GitHub)
Week 5	ATmega2560 MCU: memory, I/O ports, UART	Lab 4: SmartCAR control via Android Bluetooth
Week 6	ATmega2560 UART control & Bluetooth communication between Arduino platform and Android device	Lab 5: SmartCAR control through your own customized Android app (Due: Project proposal)
Week 7	Midterm exam	
Week 8	ATmega2560 Timer, Interrupts & Ultrasonic sensors	Lab 6: SmartCAR ultrasonic sensing
Week 9	Infrared sensors & Buzzer	Lab 7: SmartCAR infrared sensing
Week 10	Acquiring location information from Android device & line tracing	Lab 8: Implementation of line tracer
Week 11	Gyroscope, accelerometer, and compass sensors	Lab 9: Using gyroscope, accelerometer, and compass sensors
Week 12	Project	Team meeting (for progress check)
Week 13	Project	Team meeting (for progress check)
Week 14	Course wrap-up & next steps	
Week 15	Project presentation & demo I (Due: source code, presentation slides, & poster slide)	Project presentation & demo II
Week 16	Final week (no final exam)	

Lab Session

- Practice in-lab programming exercises based on the lecture materials
- Upload source codes for lab assignments in Ewha Cyber Campus after the lab session
  - Due: 11:59pm on the lab day
- Once you are done, you can leave the session after checking with me or TA
- Or, continue to work on programming for other homework assignments

Lab Policy

- 1) Please check out your SmartCAR (& Nexus 7 tablet) as soon as you arrive at the classroom
- 2) Please complete lab assignments
- 3) Upload required files to Ewha Cyber Campus
- 4) Check with me or TA
- **5) Please upload a null firmware to SmartCAR before you return it!!!**
  - **This will be a part of your lab score**
- 6) Please remove files that you created or downloaded in your computer after you are done
  - Remove your project completely
- 7) Please shut down your computer before you leave
- 8) Return the checked-out SmartCAR (& Nexus 7 tablet) to TA

NOTE: How to run SmartCAR in Lab

- Power OFF
  - Compile your code
  - Lift up your SmartCAR with your hand
  - Upload your code
- Disconnect the USB cable
- Go to find a spacious area
- Put it down there
- Power ON
- It will run your firmware
- After test, turn power OFF

Lab Announcement

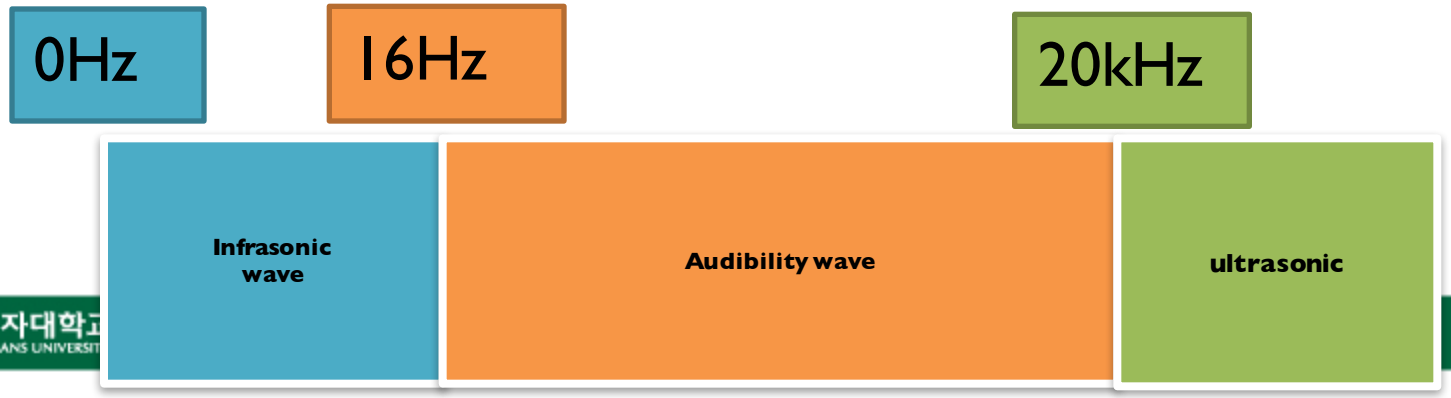
- Bluetooth pairing "headache"
  - Because there are so many Bluetooth devices in the classroom with the same name
    - "**155v2.1.7_hb**" <- SmartCAR
  - So please go outside with your SmartCAR and your Android device, and then pair them
  - Please do not pair with other students' devices

Today

- Lab announcement
- **Ultrasonic sensor**
- Lab assignment #6
- Course announcement

Ultrasonic wave

- Ultrasonic wave?
  - Sound wave with high frequency
  - Sound wave
    - Sound is transmitted through gases, plasma, and liquids
    - Audible wave
      - 20Hz~20kHz spectrum
    - Ultrasonic wave
      - Frequency spectrum where human cannot hear
  - Spectrum



SmartCAR Ultrasonic Sensors

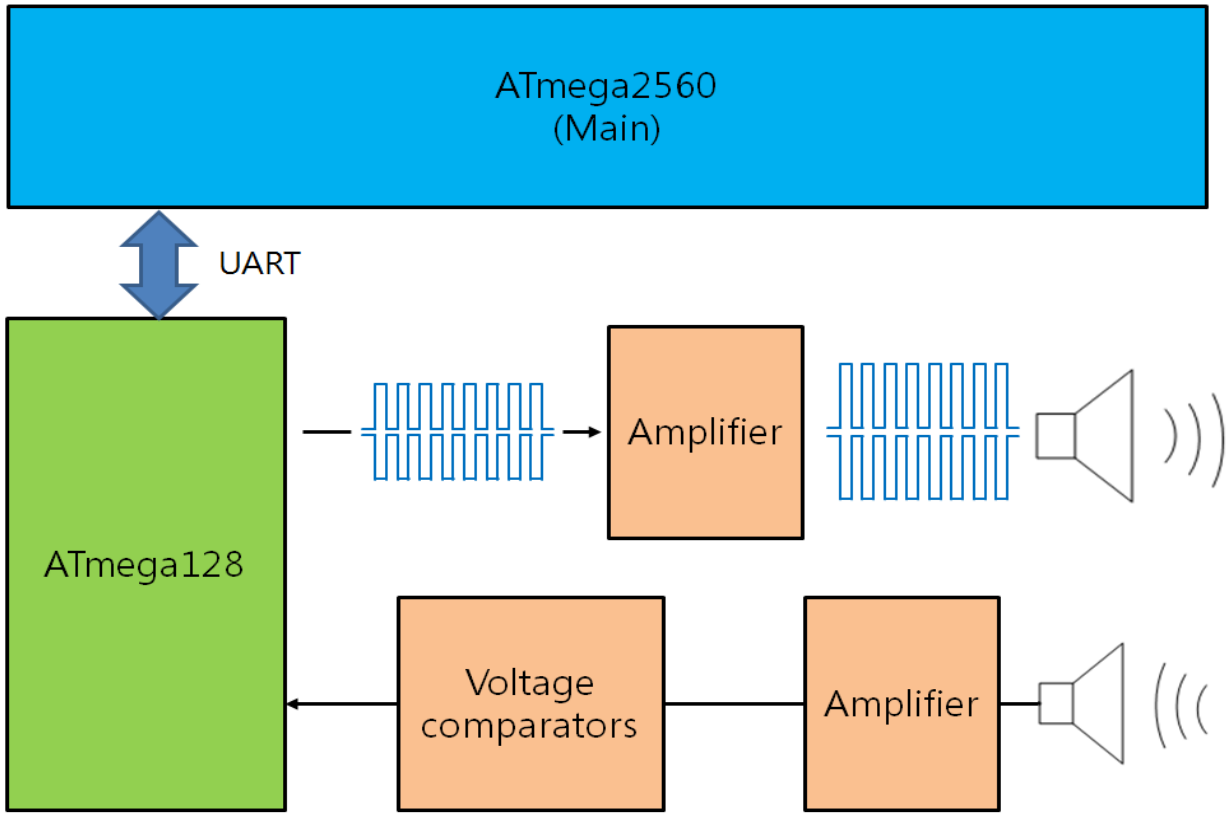
- 12 ultrasonic sensors
  - ① Ultrasonic sensor (Transmitter)

    Transmit an ultrasonic wave to detect an object
  - ② Ultrasonic sensor (Receiver)

    Receive the ultrasonic wave transmitted from TX

    => Calculates the distance from the TX-RX time diff

Ultrasonic Sensor Hardware Architecture



SmartCAR UART Port Configuration

- UART1 port is used for ultrasonic sensors

UART No.	Name	Port / Number	Etc
UART0	RXD0	PE0 / -	Program port Bluetooth port
	TXD0	PE1 / -	
UART1	RXD1	PD2 / 19	Ultrasonic sensor
	TXD1	PD3 / 18	
UART2	RXD2	PH0 / 17	Extension board 1
	TXD2	PH1 / 16	
UART3	RXD3	PJ0 / 15	Extension board 2
	TXD3	PJ1 / 14	

- Baud rate should be set to 115200bps

Main MCU and ATmega128 Communication

- OFF
  - Stop measuring from the ultrasonic

TX Data Packet (ATmega2560 -> ATmega128)				
Start		ID		CSC
0x76	0x00	0x0F	0x00	0x0F

- CSC : to check error – all ID sum & 0xFF

RX Data Packet (ATmega128 -> ATmega2560)								
Start		ID		DATA				
0x76	0x00	0x1F	0x00	0x00	0x00	0x00	0x00	0x00
DATA							CSC	
0x00	0x00	0x00	0x00	0x00	0x00	0x00		

- CSC : to check error – all ID & Data sum & 0xFF

Main MCU and ATmega128 Communication

- Basic
  - Front 3 ultrasonic sensors (F2, F3, F4) & Rear 1 ultrasonic sensor (R2)
  - Send a TX data request packet, and receive a RX data packet for the measurement continuously

TX Data Packet (ATmega2560 -> ATmega128)				
Start		ID		CSC
0x76	0x00	0x10	0x00	0x10

RX Data Packet (ATmega128 -> ATmega2560)								
Start		ID		DATA				
0x76	0x00	0x11	0x00	0x00	0x00	F2	F3	F4
DATA							CSC	
0x00	0x00	0x00	0x00	R2	0x00	0x00		

- F0~6 : distance in front ultrasonic sensors
- R0~4 : distance in rear ultrasonic sensors

Main MCU and ATmega128 Communication

- Right
  - Front 5 ultrasonic sensors (F2 ~ F6) & Rear 5 ultrasonic sensors (R0 ~ R4)
  - Send a TX data request packet, and receive a RX data packet for the measurement continuously

TX Data Packet (ATmega2560 -> ATmega128)				
Start		ID		CSC
0x76	0x00	0x20	0x00	0x20

RX Data Packet (ATmega128 -> ATmega2560)								
Start		ID		DATA				
0x76	0x00	0x21	0x00	0x00	0x00	F2	F3	F4
DATA							CSC	
F5	F6	R0	R1	R2	R3	R4		

- F0~6 : distance in front ultrasonic sensors
- R0~4 : distance in rear ultrasonic sensors

Main MCU and ATmega128 Communication

- Left
  - Front 5 ultrasonic sensors (F0 ~ F4) & Rear 5 ultrasonic sensors (R0 ~ R4)
  - Send a TX data request packet, and receive a RX data packet for the measurement continuously

TX Data Packet (ATmega2560 -> ATmega128)				
Start		ID		CSC
0x76	0x00	0x30	0x00	0x30

RX Data Packet (ATmega128 -> ATmega2560)								
Start		ID		DATA				
0x76	0x00	0x31	0x00	F0	F1	F2	F3	F4
DATA								CSC
0x00	0x00	R0	R1	R2	R3	R4		

- F0~6 : distance in front ultrasonic sensors
- R0~4 : distance in rear ultrasonic sensors

Main MCU and ATmega128 Communication

- Front
  - Front 7 ultrasonic sensors (F0 ~ F6)
  - Send a TX data request packet, and receive a RX data packet for the measurement continuously

TX Data Packet (ATmega2560 -> ATmega128)				
Start		ID		CSC
0x76	0x00	0x40	0x00	0x40

RX Data Packet (ATmega128 -> ATmega2560)								
Start		ID		DATA				
0x76	0x00	0x41	0x00	F0	F1	F2	F3	F4
DATA							CSC	
F5	F6	0x00	0x00	0x00	0x00	0x00		

- F0~6 : distance in front ultrasonic sensors
- R0~4 : distance in rear ultrasonic sensors

Main MCU and ATmega128 Communication

- Back
  - Front 2 ultrasonic sensors (F0, F6) & Rear 5 ultrasonic sensors (R0 ~ R4)
  - Send a TX data request packet, and receive a RX data packet for the measurement continuously

TX Data Packet (ATmega2560 -> ATmega128)				
Start		ID		CSC
0x76	0x00	0x50	0x00	0x50

RX Data Packet (ATmega128 -> ATmega2560)								
Start		ID		DATA				
0x76	0x00	0x51	0x00	F0	0x00	0x00	0x00	0x00
DATA							CSC	
0x00	F6	R0	R1	R2	R3	R4		

- F0~6 : distance in front ultrasonic sensors
- R0~4 : distance in rear ultrasonic sensors

Main MCU and ATmega128 Communication

- All
  - Front 7 ultrasonic sensors (F0 ~ F6) & Rear 5 ultrasonic sensors (R0 ~ R4)
  - Send a TX data request packet, and receive a RX data packet for the measurement continuously

TX Data Packet (ATmega2560 -> ATmega128)				
Start		ID		CSC
0x76	0x00	0xF0	0x00	0xF0

RX Data Packet (ATmega128 -> ATmega2560)								
Start		ID		DATA				
0x76	0x00	0xF1	0x00	F0	F1	F2	F3	F4
DATA							CSC	
F5	F6	R0	R1	R2	R3	R4		

- F0~6 : distance in front ultrasonic sensors
- R0~4 : distance in rear ultrasonic sensors

SmartCAR Firmware

```
#define NUM_TX_BYTES        5
#define NUM_RX_BYTES        17

unsigned char TX_buf[NUM_TX_BYTES] = {0x76, 0x00, 0xF0, 0x00, 0xF0};
unsigned char TX_stop_buf[NUM_TX_BYTES] = {0x76, 0x00, 0x0F, 0x00, 0x0F};
unsigned char RX_buf[NUM_RX_BYTES];

boolean  ultrasonic_result  = false;

void setup()
{
    int i = 0;
    Serial.begin(115200);

    while (text[i] != '₩0')
        Serial.write(text[i++]);

    Serial.write("Received  cmds: ");

    Serial1.begin(115200);
    //initialize  ports
    pinMode(....);
    ....
    digitalWrite(...);
}
```

SmartCAR Firmware

```
void loop()
{
}

void serialEvent()
{
    int command = Serial.read();

    switch (command)
    {
        case 1:
            move_stop();
            delay(500);

            move_forward();
            break;
        case 2:
            move_stop();
            delay(500);

            turn_left();
            break;
        case 3:
            move_stop();
            delay(500);

            turn_right();
            break;
        case 4:
            move_stop();
            delay(500);

            move_backward();
            break;
        case 5:
            move_stop();
            break;
        case 6:
            front_led_control(true);
            break;
        case 7:
            front_led_control(false);
            break;
        case 8:
            rear_led_control(true);
            break;
        case 9:
            rear_led_control(false);
            break;
        case 10:
            ultrasonic_sensor_read();
            break;
        default:
            move_stop();
            front_led_control(false);
            rear_led_control(false);
    }
}
```

SmartCAR Firmware

```
void ultrasonic_sensor_read()
{
    ultrasonic_result = false;
    Serial1.write(TX_buf, NUM_TX_BYTES);
}

void serialEvent1()
{
    unsigned char z, tmp = 0;
    Serial1.readBytes((char *)RX_buf, NUM_RX_BYTES);

    if ( (RX_buf[0] == 0x76) && (RX_buf[1] == 0x00) &&
         (ultrasonic_result == false) )
    {
        for (z = 2; z < NUM_RX_BYTES-1; z++)
            tmp += RX_buf[z];

        tmp = tmp & 0xFF;

        if (RX_buf[NUM_RX_BYTES-1] == tmp)
        {
            Serial.println("FRONT");
            for (z=4; z < 11; z++)
            {
                Serial.print(" F");
                Serial.print(z-4);
                Serial.print(": ");
                Serial.print(RX_buf[z]);
            }
            Serial.println("₩nBACK");
            for (z=11; z < NUM_RX_BYTES-1; z++)
            {
                Serial.print(" B");
                Serial.print(z-11);
                Serial.print(": ");
                Serial.print(RX_buf[z]);
            }
        }
        ultrasonic_result = true;
        Serial1.write(TX_stop_buf,
                      NUM_TX_BYTES);
    }
}
```

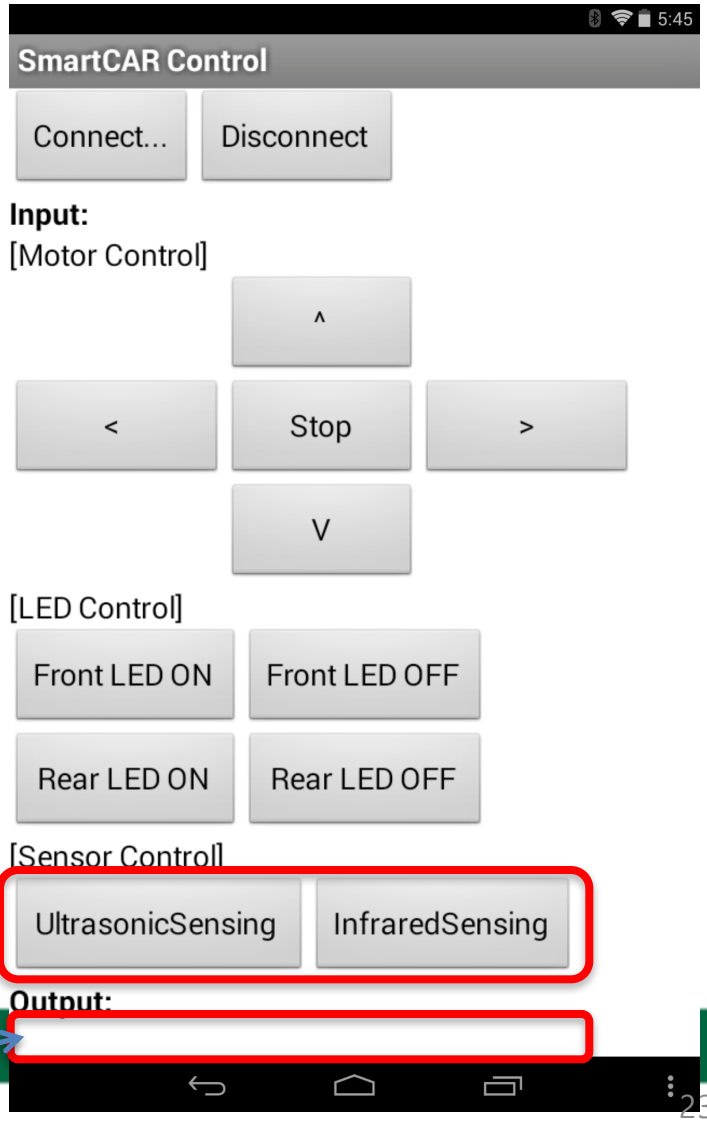
Execute the ultrasonic sensor!

Measure only once and then disable the ultrasonic sensor!

SmartCAR Control App

- http://ai2.appinventor.mit.edu
- Click on "New Project"
- Enter "SmartCAR_Ctr" in Project Name (One word, no space)

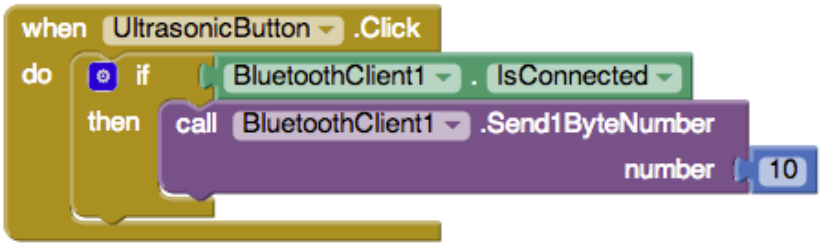
- Under "User Interface"
  - Drag-and-drop "ListPicker" component
    - To select a Bluetooth device
  - Drag-and-drop "Button" component
  - Drag-and-drop "Label" component
- Under "Layout"
  - Drag-and-drop "TableArrangement"
  - Drag-and-drop "TableArrangement"
- Under "Connectivity"
  - Drag-and-drop "BluetoothClient"
    - **Uncheck "Secure"**
- Under "Sensors"
  - Drag-and-drop "AccelerometerSensor"



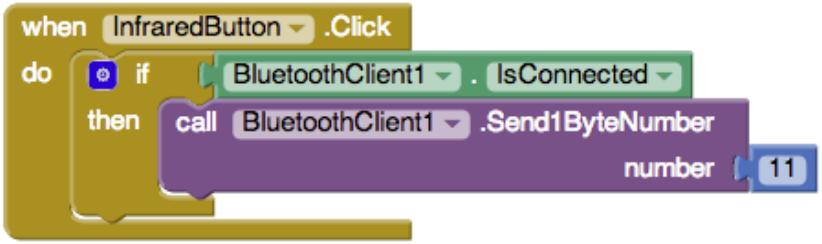
BTOutput (Label)

SmartCAR Sensor Control

- Sensing Ultrasonic sensor (command byte: 10)
  - Send "10" in number using "BluetoothClient.Send1ByteNumber"

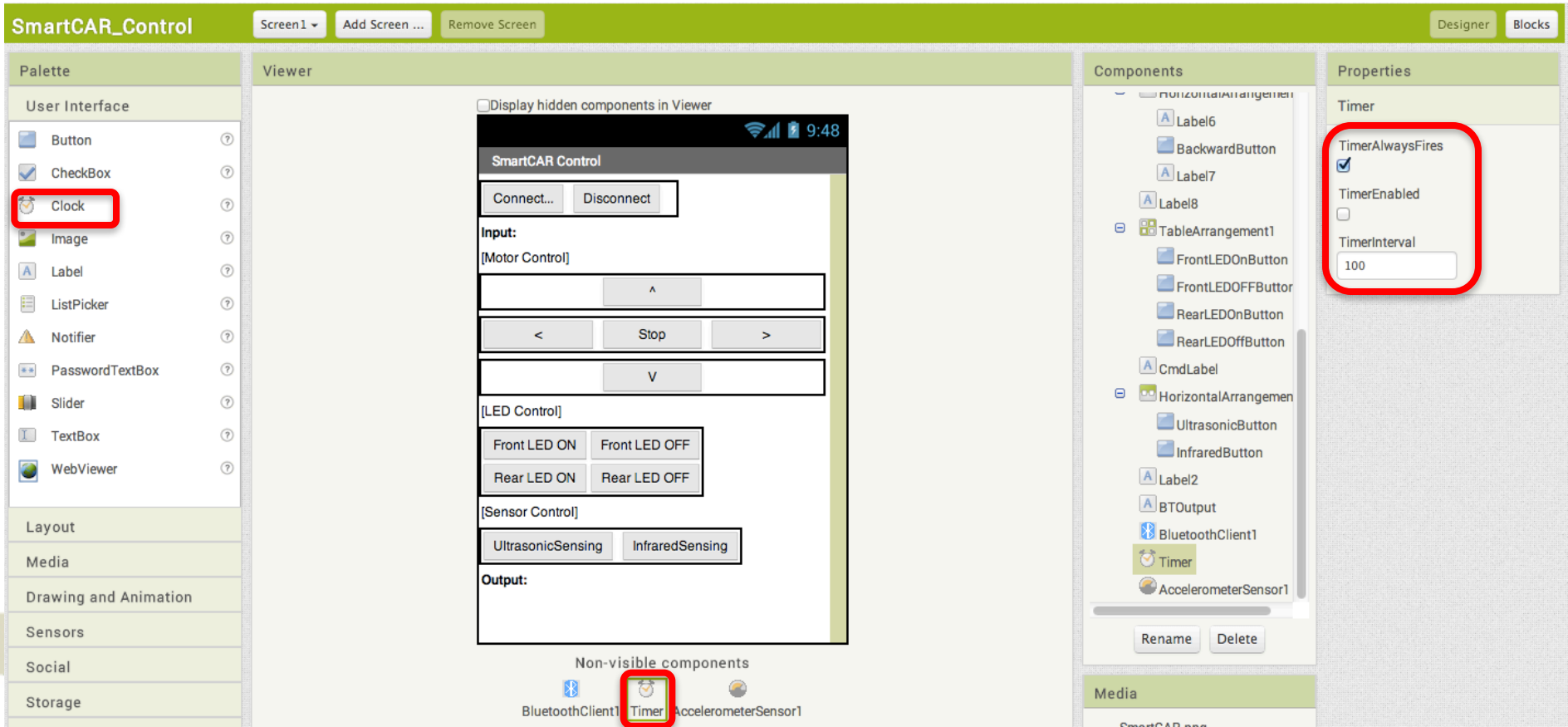


- Sensing Infrared sensor (command byte: 11)
  - Send "11" in number using "BluetoothClient.Send1ByteNumber"



Receiving Bytes from SmartCAR

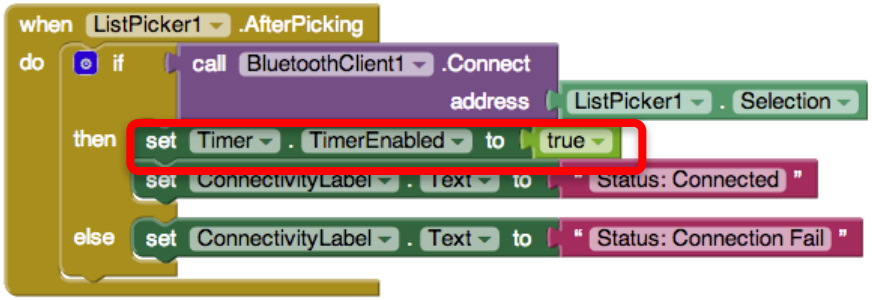
- Use "Clock" component: timer functionality
  - To periodically check whether any data have been received from Bluetooth
  - TimerAlwaysFires? : to select whether to expire periodically or not
  - TimerEnabled : to select whether to initially enable or not

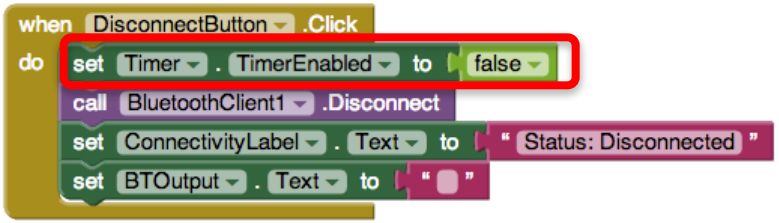


Receiving Bytes from SmartCAR

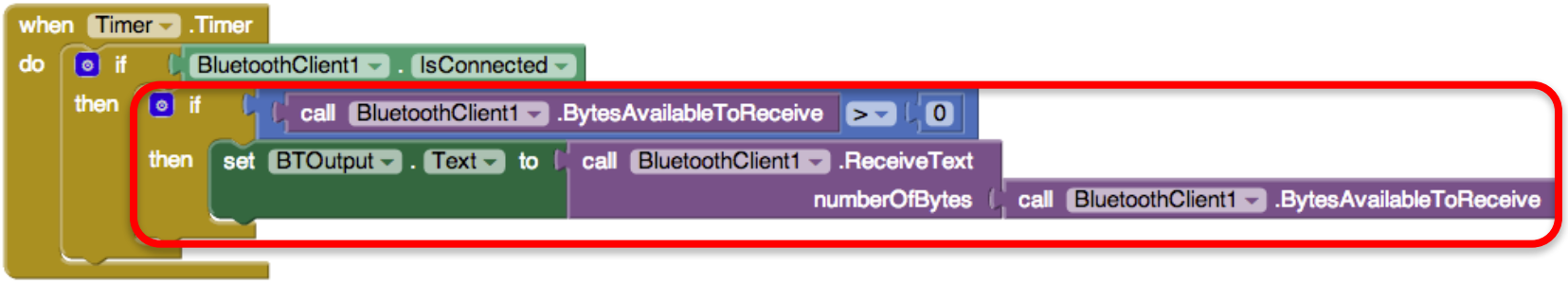
- 1) When Bluetooth is connected, then the timer is enabled
- 2) When "Timer" expires, check if there are received data
  - If so, print them to a label

1)





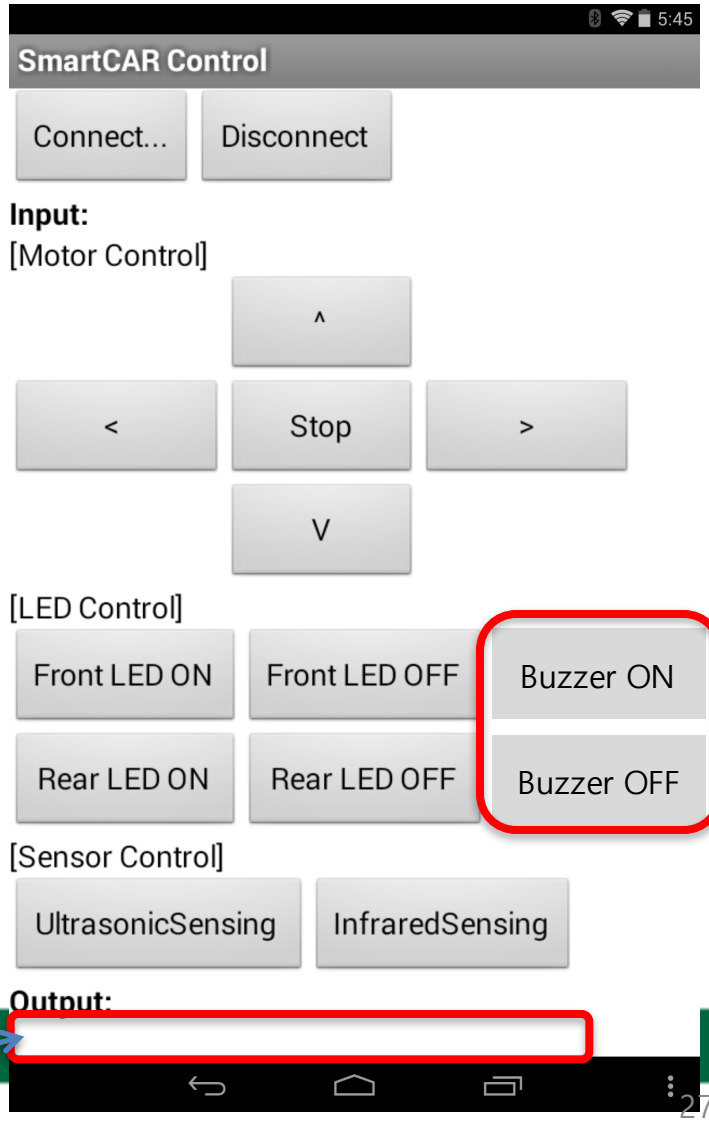
2)



SmartCAR Control App with Buzzer

- http://ai2.appinventor.mit.edu
- Click on "New Project"
- Enter "SmartCAR_Ctr" in Project Name (One word, no space)

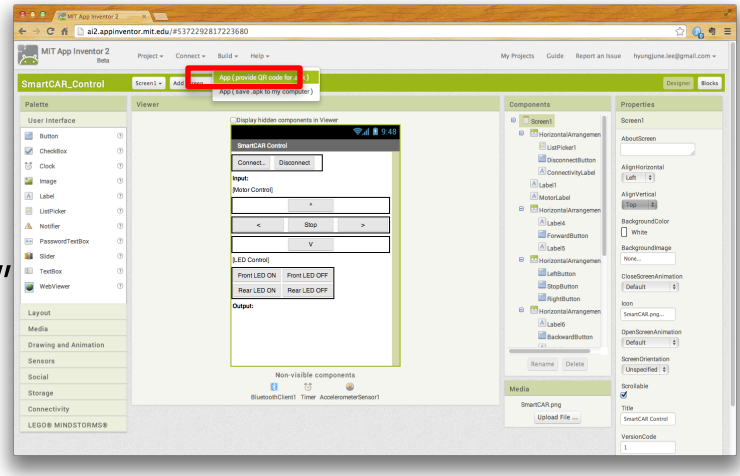
- Under "User Interface"
  - Drag-and-drop "ListPicker" component
    - To select a Bluetooth device
  - Drag-and-drop "Button" component
  - Drag-and-drop "Label" component
- Under "Layout"
  - Drag-and-drop "TableArrangement"
  - Drag-and-drop "TableArrangement"
- Under "Connectivity"
  - Drag-and-drop "BluetoothClient"
    - **Uncheck "Secure"**
- Under "Sensors"
  - Drag-and-drop "AccelerometerSensor"

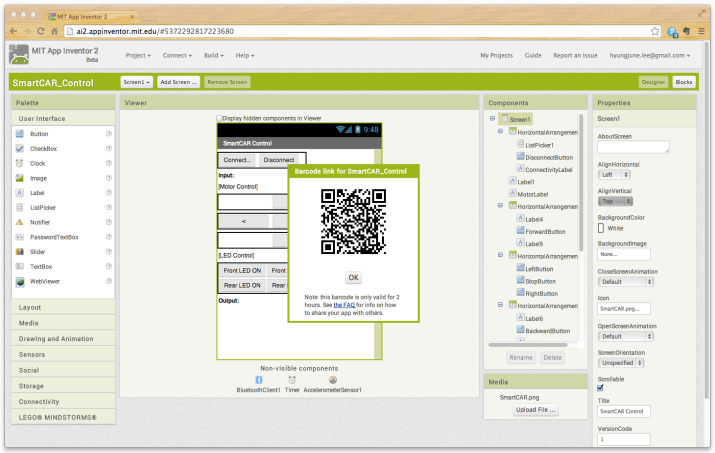


BTOutput (Label)

How to Run your Android app

- 1. Install “QR Barcode Scanner” in Play Store (Android device)
- 2. Click on “Build” (PC)
  - Click on “App (provide QR code for .apk)”
- 3. Run “QR Barcode Scanner” (Android device)
- 4. Touch the URL link (Android device)
- 5. Select “OK” to install .apk file (Android device)





Lab Assignment #6: Ultrasonic sensor & Buzzer

- Submit three following files to Cyber Campus
  - 1) lab6.cpp (Arduino firmware code)
  - 2) lab6.h (Arduino firmware code)
  - 3) SmartCAR_Ctr.apk (Android app package)
    - You should set the app icon image to "SmartCAR.png"
    - In App Inventor,
      "Build" ➔ "App (save .apk to my computer)"

- Show your result to TA or instructor