S & M 2412

Development of Arduino-Yun-based JavaScript Logic Expression Programmable IoT for Automation System

Pratikto,[1] Raydha Zul Fitriani,[2] and Yean-Der Kuan[2*]

[1]Department of Refrigeration and Air Conditioning, Politeknik Negeri Bandung,
Jl. Gegerkalong Hilir, Ds. Ciwaruga, Bandung 40559, Jawa Barat, Indonesia
[2]Department of Refrigeration, Air-Conditioning and Energy Engineering,
National Chin-Yi University of Technology,
No. 57, Sec. 2, Zhongshan Rd., Taiping Dist., Taichung City 41170, Taiwan

(Received June 15, 2020; accepted November 18, 2020)

Keywords: Arduino Yun, web programming, JavaScript, IoT, DAQ USB

In this paper, a web-based application user interface (UI) to control Arduino input/output (I/O) through Hypertext Markup Language (HTML) and JavaScript logic expressions is developed. A simple server can be accessed from the internet and run on Arduino to support communication in controlling I/O, and is easier to operate as an IoT device. Arduino Yun is an option if the HTML file is sufficiently large because it is equipped with double processors and an SD card slot. Some scripts, namely, combination of OR and AND (ORN) direct on line (DOL), equivalent to start/stop or set/reset logic, as well as comparator, timer, and counter functions in one program line, are defined and then the command is equivalent to a programmable logic controller (PLC) ladder diagram rung. Arduino Yun is tested as a logic controller and applied to a virtual plant built using LabVIEW. The communication between the virtual plant in LabVIEW and Arduino Yun uses a NI DAQ 6009 USB device. The implementation results show that the input signal after undergoing one loop to give a signal to the virtual plant experiences a delay of 0.3 s, and this delay also occurs with the timer. If analog signals are included, the delay is 0.53 s.

1. Introduction

Microcontrollers such as Arduino(1) are widely applied in many fields, from the academe to industry.(2–6) Many researchers have used Arduino as a microcontroller in their research.(7,8) Arduino is produced in various types, and each type or a combination of various types can be selected to support a particular experiment, as explained by Louis, in whose paper various Arduino boards were described as well as their implementation.(9) The use of Arduino as a data acquisition tool has been supported by PLX-DAQ, a program based on Microsoft Excel.(10) Many libraries have also been developed for users to utilize Arduino for specific purposes.(11) MathWorks also provides library or connection tools for MATLAB and Arduino.(12) Arduino can be programmed similarly to programmable logic controller (PLC) with LDmicro.(13) Microsoft also allows Windows 10 to be used as a virtual shield from Arduino.(14)

[*]Corresponding author: e-mail: ydkuan@ncut.edu.tw

https://doi.org/10.18494/SAM.2020.3150

ISSN 0914-4935 © MYU K.K.
https://myukk.org/

Arduino is a multipurpose device; thus, it can be used in various designs from basic learning to internet-based applications such as the Internet of Things (IoT). Mahalakshmi and Vigneshwaran proposed home automation with Arduino as the heart of the system. The system used various sensors and could be controlled using an Android smartphone app or web application. It was also equipped with Bluetooth and GSM modules.(15) Hajjaj *et al.* developed an autonomous floodgate through the IoT and used Raspberry Pi to collect the data, which were then published to the cloud.(16) Arduino is also coupled with other devices such as mini PCs including Raspberry Pi.(17) Many types of sensors can be connected to Arduino. Praveena *et al.* used ultrasonic sensors and Arduino to make a smart railway gate. The system also used a servo motor as an actuator for the gate and an ESP 32 cam to detect any obstacles on the railway.(18) Libraries in control, for example, PID(19) as well as fuzzy logic libraries,(20) can be found. In particular, one type of Arduino, namely, Arduino Yun, is equipped with two processors: one is an ordinary Arduino program processor and the other runs Linux to manage wireless or web-based communication.(21)

A JavaScript library component that can be used to 'talk' to Arduino more easily via the Firmata protocol (a standard protocol for computers communicating with hardware) has been created. The library is called Johnny-Five, which is a JavaScript library that is designed to talk to Arduino, Raspberry Pi, and other devices.(22) Johnny-Five code is executed by any device capable of supporting Node.js. However, Arduino does not run any JavaScript programs. The advantages of using Arduino are the low cost(23) and the ability to obtain a simple interface device.(24)

A fundamental process when utilizing Arduino is managing and operating the functions of its input/output (I/O). The state value of each pin from Arduino can be controlled via a PC, a tablet, or a smartphone running Hypertext Markup Language (HTML)-JavaScript with client-server communication. With this simple step, commands can be sent from the client and then accepted by Arduino Yun as a server, then the Arduino internal program runs the command and returns the results to the client.

The submission of commands from the client through accessing the address with the URL produces a text response when formatted as a web page that will appear in a browser page. Refreshing can be avoided and only data can be accepted with the XMLHttpRequest command. However, not all browsers can generate a response to this command if it is run locally on the client.

To solve this problem, the web page code can be sent from the Arduino Yun SD card. One of the advantages of Arduino Yun is that the formed web server is handled by a separate processor from the processor that executes the program commands.

Flexible applications are needed to take advantage of Arduino in terms of operation, connection, function, and convenience, including flexibility in modifying the program. This type of application is very suitable when built using web programming languages such as HTML and JavaScript. In connection with these requirements, this paper focuses on the development of a logic programming application for Arduino Yun I/O controllers based on HTML and JavaScript. Programs that run in a microcontroller generally cannot be changed in run time. This is a problem or obstacle in creating a flexible control program without creating or changing the original program. Here, we discuss how to overcome the problem.

In general, IoT is used for measurement and monitoring, so the method proposed in this paper provides the ability to carry out programming. A user interface (UI) was built to translate infix to postfix instructions that can be executed by Arduino Yun in order to respond to the commands in Boolean JavaScript expressions that involve NOT, AND, and OR logic along with the timer, counter, and comparator functions.

Other elements of the program that can be used to form an OR-AND logic combination, namely, combination of OR and AND (ORN), are defined. The ORN command is the basis for establishing a direct on line (DOL) equivalent to start/stop or set/reset logic.

LabVIEW is software from National Instruments that can be used to operate or program other devices.(25) Arduino with LabVIEW can be connected using serial communication. It will be better if communication using LabVIEW can be performed with Modbus(26) or a web server.(27) The performance of Arduino Yun as a logic controller was tested on virtual plant control in LabVIEW with a NI DAQ 6009 USB as a communication device.

The method developed in this paper provides a means of implementing control algorithms on hardware in an effective way on targets with simple preparation.

2. Arduino Yun I/O Controller Logic Script

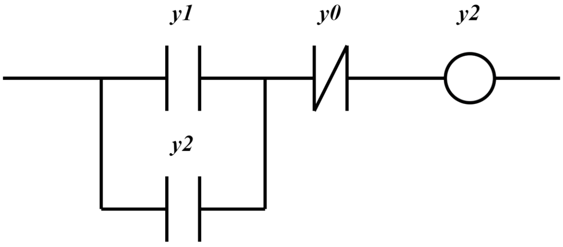
If a PLC control program is written in the form of a ladder diagram, the control program for Arduino Yun will be in the form of a logic script written with JavaScript syntax. Each line of the script can be considered to correspond to a rung in the ladder diagram. Figure 1 shows on/off or start/stop control logic in the ladder diagram and equivalent script forms.

Logic expression operators are available in JavaScript. Converting a rung ladder to JavaScript does not have to use an existing script or even have to define the required operators. The basic elements involved in control logic are a timer, a counter, and a comparator, so the functions associated with these elements must be formed. The symbols for the timer, counter, and comparator operators are written as follows:

$$y0 = T(n, d, v);$$

$$y1 = Cmp(v1, \text{“>”}, v2);$$

$$y3 = C(n, count, v);$$



Logic Script
y2 = (y2||y1)&&!y0

Fig. 1. Start/stop JavaScript logic equivalent with ladder diagram.

n represents the index of the logical element, *d* is the time when the timer is active (on), and *v* is the discrete quantity used to activate the timer. Comparators *v*1 and *v*2 are analog or discrete values compared with a specified operation. The *d* value in the counter element represents the index, *count* represents the count when the counter is on, and *v* is the counted input signal.

The detailed codes for the timer, counter, and comparator are shown as follows.

2.1 UI

```
//Reset Timer
bool TR(int n,bool v){
 if (v) TimerInit[n]=0;
 return v; }
//Object
void initTimer(){
for(int k=0;k<jumTimer;k++)
TimerInit[k]=0;//inisialisasi
timer
}
//Process Object
bool T(int n,float d,bool v){
 tsecond = millis();
if (v==true){
   if (TimerInit[n]==0) {
   d1[n]=millis();
   TimerInit[n]=1;
   }
}
 unsigned long  t=millis()-d1[n];
if (t<=d*1000) valTimer[n]=(millis()-d1[n])/1000;
if (t>d*1000){
outTimer[n]=v;valTimer[n]=d;}
else {outTimer[n]=false;}
//document.getElementById(“demo2”).innerHTML
=TimerInit[n];
return outTimer[n];
}
```

```
//Comparator
bool Cmp(int Av, char op, int Am){
  bool ret = false;
  switch(op) {
   case ‘=’ :
if (Av == Am)
  ret = true;
    else
 ret = false;
    break;
    case ‘<’ :
 if (Av < Am)
  ret = true;
 else
  ret = false;
 break;
    case ‘>’ :
 if (Av > Am)
  ret = true;
 else
  ret = false;
 break;
    default :
 ret = false;
    }
    return ret;
}
```

```
//Counter
void CounterInit(){
  int i;
  for( i = 0;i<jumlahCounter;i++) {
   vlast[i] = false;
   Cs[i] = 0;
   Cv[i]=0; }
}
bool Cnt(int n, int cnt, bool v){
  bool ret;
  if ((Cs[n] < cnt) && (v!=vlast[n]) && v) {
    Cs[n] = Cs[n] + 1;
    Cv[n] = Cs[n];
  }
  ret = false;
  if ((Cs[n] >= cnt) ) { ret = true;}
  vlast[n] = v;
  //Serial.println(Cs[n]);
  return ret;
}
bool CR(int n, bool v){
   if (v)  Cs[n] = 0;
 return v;
  }
```

The aim of a UI is to make programming easier; therefore, from the display side, it is better to only show the important tools on the page. Indeed, the native way to give or read the value of a pin must be preceded by at least two steps: the first is determining whether a port is an input or an output and the second is determining its value. Such steps are unnecessary through this UI because they can be carried out automatically by reading the lines of the program logic used. Thus, programming can be made easy.

In fact, when a program is running, there is a main program that is always repeated. With this UI, programs are only written from the key of their main logic. This is different from the usual script programming.

The UI for programming hardware usually does not provide a simulator, and if there is one, only part of the function can be simulated. This built UI provides facilities for simulating the program to see whether the results of the logic are implemented as desired.

Figure 2 shows a UI design that is implemented using HTML and JavaScript code so that it has three functions: a script editor, a simulator, and a translator of the program logic that will be run by Arduino Yun.

In the diagram, the UI appears to be simple, but it is not simple because as a text editor, the UI checks whether the syntax is in accordance with the required equation format. In fact, the role of the UI as a text editor is inseparable from its role as a translator because the syntax test is also carried out to translate the equations. The importance of the translation here is that JavaScript is not run on the Arduino processor, but instead the line-by-line logic of the instructions is executed.

Equations are written in infix form. The infix form contains logic that cannot be worked through steps that rely on memory and its input and output processes. Thus, it must be translated into a form that can be worked on step by step, namely, in postfix form. The UI performs this translation, so the script that the programmer writes in infix form is sent to the controller in postfix form.

Outline of algorithm:

i. Read the infix code equation text.

ii. Check the equation. If it is incorrect, an error alert will be issued. If it is correct, continue to the next stage.

iii. If simulation is selected, perform the simulation. If not, go to the next stage.

iv. Translate from the infix form to the postfix form.

v. Send the postfix form to Arduino.

The infix-to-postfix conversion algorithm is not executed in Arduino Yun.

Stage iv is the most crucial stage, because it contains details of the following complex steps that are not included in the standard algorithm:

1. Determine the terms in the equation.
2. Check and record the variables and numbers used.
3. Define the variable as an input or an output.

These three steps must be carried out when implementing the standard infix-to-postfix conversion algorithm so that Arduino can perform the job more smartly, because some tasks that are always performed in the native way are not necessary, as previously explained. The algorithm is shown as a flowchart in Fig. 3. The UI is connected to the Arduino Yun web service. In the first connection, the credential is required as it is a secure session.

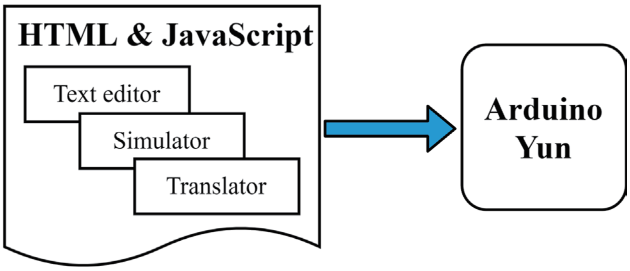


Fig. 2. (Color online) UI structure of Arduino Yun I/O programmer.

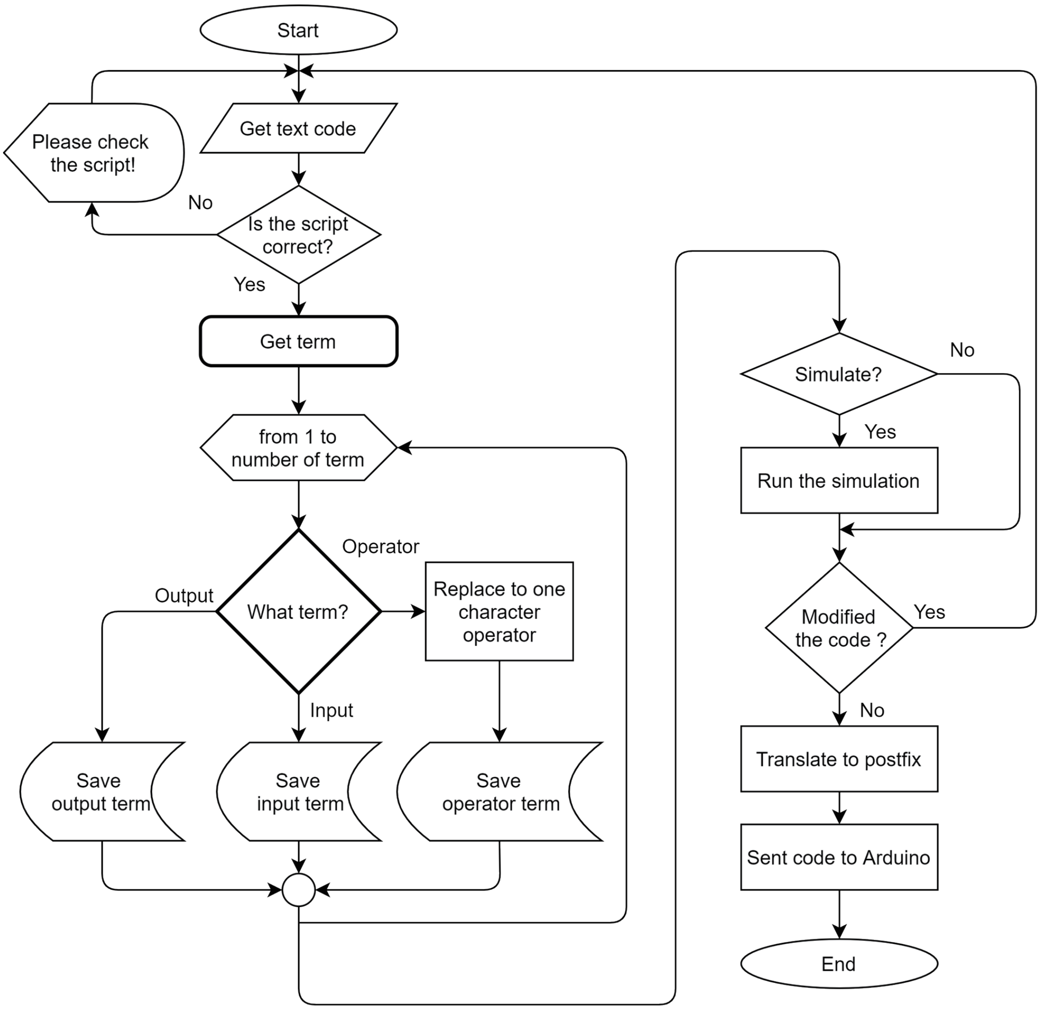


Fig. 3. Flowchart of UI program.

2.2 Arduino side code

The logical expression is translated line by line by the UI into postfix form, which can be calculated and processed by the program in Arduino Yun to continue giving or reading the pin value or also save it as a value of the variable used. In this way, Arduino Yun becomes a programmable logic control device and works in a standalone manner based on JavaScript expressions. Figure 4 shows an outline of how Arduino Yun processes the translated expression. A special logic can be seen in the flowchart, where the pins used as the input will be automatically assigned according to the translation by the UI.

2.3 Virtual plant

The performance of a controller must be tested in the complete situation as the control algorithm is applied. There are five elements that must be present, namely, the set point value,

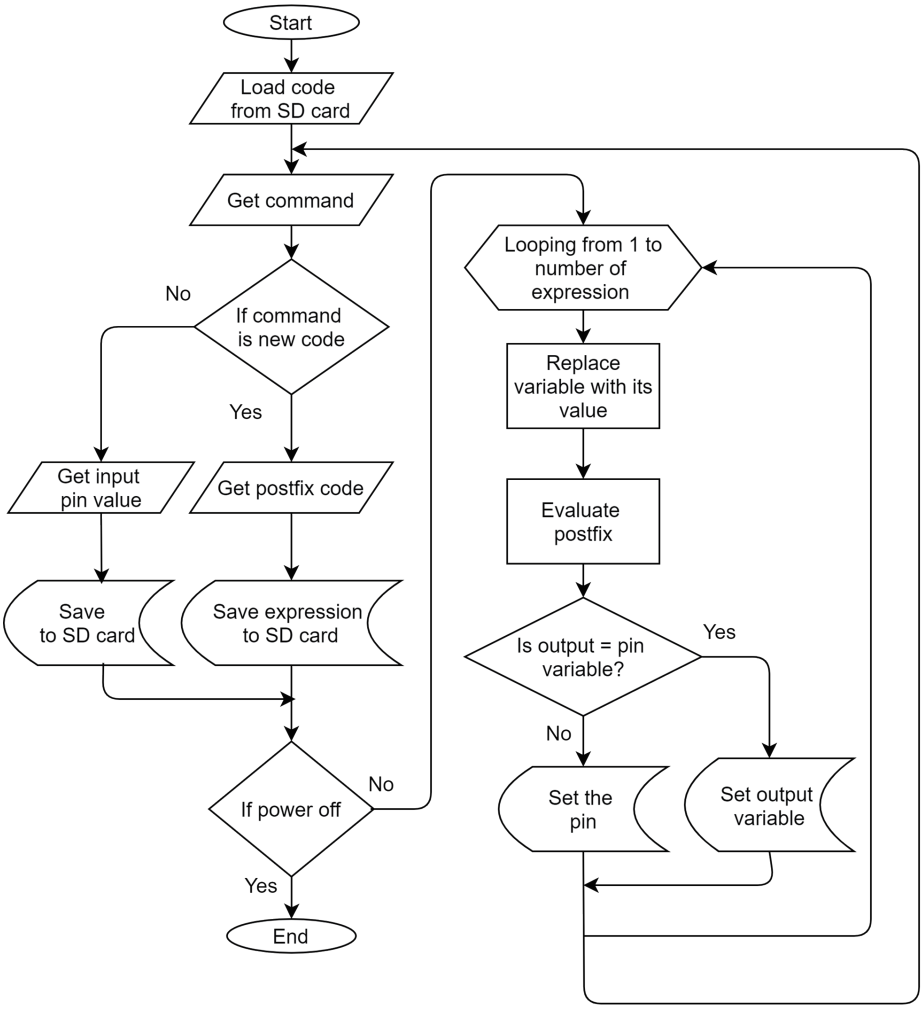


Fig. 4. Flowchart of processing script code in Arduino Yun.

feedback, controller, actuator, and plant. In this paper, the test of the controller focuses on the correctness of the logic and the signal speed, so the plant's behavior is not designed specifically and the plant is selected in a virtual form built using LabVIEW. Using this virtual plant, the situation of the system becomes a hardware in the loop (HIL), which means that sensors and actuators also become virtual, that is, they are replaced by a direct connection to the values issued by the virtual plant through the interfacing device and controller.

Assuming that the plant type is also not focused on, a plant in the form of a water tank system was chosen to be virtually built in LabVIEW to test Arduino Yun as a controller programmed in the proposed manner. The system variable is the level of water controlled by turning on and off the incoming water flow. The bottom of the tank is made perforated as an outlet. Suppose that the system has an input channel to turn on and off the incoming water flow, a level sensor and an output channel indicating the specified highest and lowest water levels, respectively, are shown in Fig. 5.

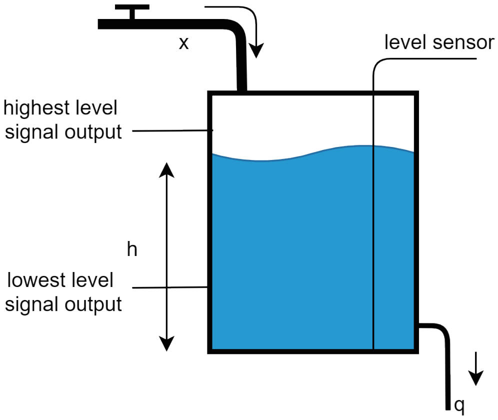


Fig. 5. (Color online) Water level in tank plant.

The plant is modeled using the first-order I/O equation as

$$h(t_i) = h(t_{i-1}) + (x - q)T / 1000, \tag{1}$$

where h is the water level, and t_i and t_{i-1} are the time after and the time before the controller signal is given, respectively. Moreover, x, q, and T are the input and output flow rates of the tank, and the sampling time in milliseconds, respectively. Figure 6 shows a block diagram of the system, which includes the virtual plant and its communication interface used to test the performance of Arduino Yun as a controller.

3. Implementation and Test Results of I/O Arduino Yun Control

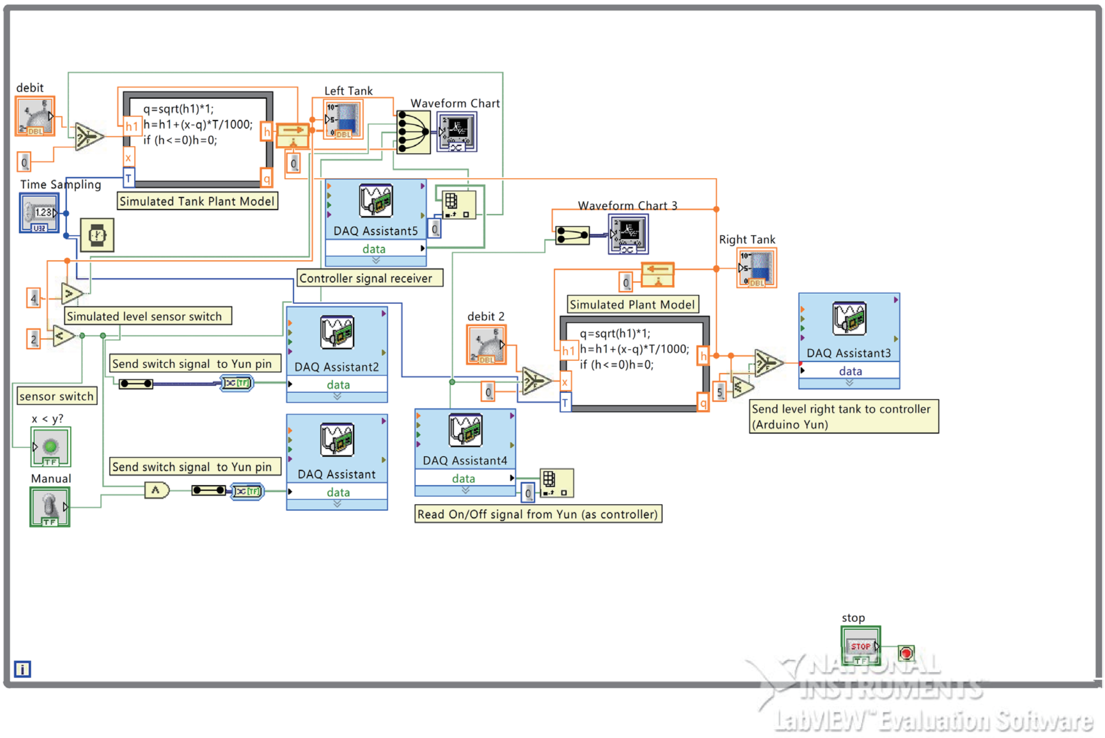
If input variables *y*4 and *y*3 are associated with *pin*4 and *pin*5, respectively, and the output is associated with *pin*3, the start stop/DOL command when written in a logic script in the UI editor is as follows:

Program 1:

```
//input Conf.
y4 = pin4; y3 = pin5;
//Rung
y5 = (y5||y1)&&!y4;
//Output
pin3 = y5;
```

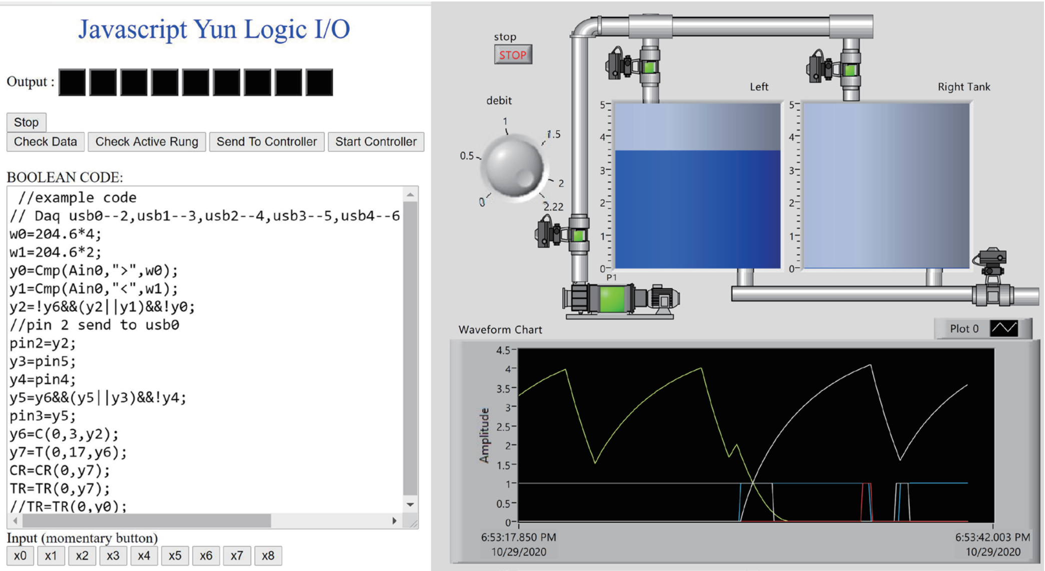
The I/O control script is written according to the provisions, with the internal variables named *y* followed by a number of up to 15. Variables involving digital pins are written with the pin name followed by a number from 0 to 13. The logic written in the script includes And (&&), Or (||), and Not (!). Every symbol used is in accordance with what is applicable to JavaScript.

Figure 7 shows the implementation of Arduino Yun I/O control based on Boolean JavaScript expressions to control the water level in a virtual tank modeled in LabVIEW.


debit
Left Tank
Waveform Chart
Simulated Tank Plant Model
Time Sampling
DAQ Assistant5
data
Controller signal receiver
Waveform Chart 3
Right Tank
Simulated Plant Model
Simulated level sensor switch
debit 2
DAQ Assistant3
data
Send switch signal to Yun pin
DAQ Assistant2
data
sensor switch
x < y?
Send level right tank to controller (Arduino Yun)
DAQ Assistant4
data
Send switch signal to Yun pin
DAQ Assistant
data
Manual
Read On/Off signal from Yun (as controller)
stop
NATIONAL INSTRUMENTS
LabVIEW™ Evaluation Software

```
q=sqrt(h1)*1;
h=h1+(x-q)*T/1000;
if (h<=0)h=0;
```

Fig. 6. (Color online) Block diagram of virtual plant.



BOOLEAN CODE:

```
 //example code
// Daq usb0--2,usb1--3,usb2--4,usb3--5,usb4--6
w0=204.6*4;
w1=204.6*2;
y0=Cmp(Ain0,">",w0);
y1=Cmp(Ain0,"<",w1);
y2=!y6&&(y2||y1)&&!y0;
//pin 2 send to usb0
pin2=y2;
y3=pin5;
y4=pin4;
y5=y6&&(y5||y3)&&!y4;
pin3=y5;
y6=C(0,3,y2);
y7=T(0,17,y6);
CR=CR(0,y7);
TR=TR(0,y7);
//TR=TR(0,y0);
```

Fig. 7. (Color online) Testing the implementation of Arduino Yun I/O programming.

As shown in Fig. 7, the levels of water in two tanks are controlled. The left tank water level is controlled by the logic script described by program 1. The *pin*3 and *pin*4 of Arduino Yun are set as the control inputs connected to the lowest and highest water level signals, respectively, while *pin*5 is set as the control output connected to the input channel of the left tank. The right tank is controlled by a script involving the comparator element where the water level signal is connected to Arduino Yun analog *pin*0 while *pin*2 is connected to the input channel of the right tank. As shown in Fig. 7, the tanks are filled alternately. The control logic script is chosen so that it involves the counter and timer elements as a whole. On the basis of the logic script programmed in Arduino Yun before the counter is true, the right tank is filled first after the counter is false, then the left tank is filled and controlled. A timer is used to reset the counter. The counter parameter determines the number of times the left tank has been filled and the timer parameter determines how long the incoming water flow of the right tank has been activated.

The virtual tank and Arduino are connected via an NI DAQ 6009 USB. The Arduino Yun pins involved are connected directly to the NI DAQ 6009 USB pins as shown in Fig. 8.

The control response is shown in Fig. 9. The water level of the left tank is regulated by Arduino Yun based on the signal given by LabVIEW, which is transmitted by the DAQ USB at *pin*3 and *pin*4 to produce a control signal at *pin*5, which is received back in LabVIEW. From the graphical data, the I/O cycle is found to take a maximum of 0.3 s. A delay of this magnitude also occurs with the timer.

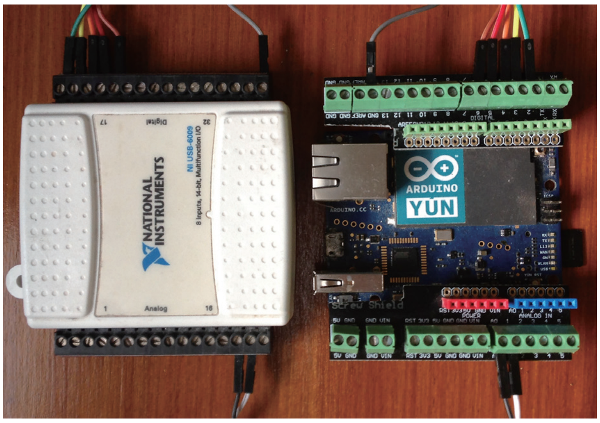


Fig. 8. (Color online) Direct connection of Arduino Yun pins and NI DAQ 6009 USB.

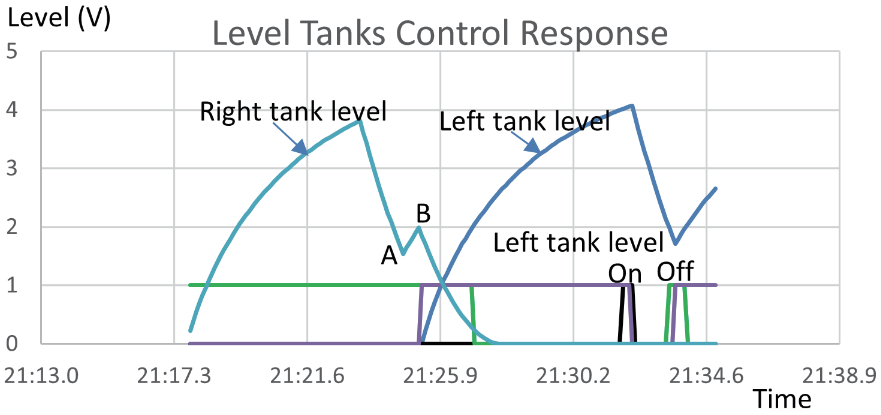


Fig. 9. (Color online) Tank level control response.

The cycle time when an analog signal is involved can be calculated from the response of the right tank water level. A command to increase the water level of the right tank from Arduino Yun is received by the virtual plant in LabVIEW at the time of point A, but in the next cycle, Arduino Yun sends the command to stop the filling as the counter setting has been reached and received by the virtual plant at the time of point B. Therefore, one cycle of communication between the virtual plant and Arduino Yun as the controller including the analog signal does not exceed the time from point A to point B, which is 0.53 s according to the graph. Thus, engaging the analog signals requires a longer cycle time. The delay that occurs is relatively small and naturally occurs because Arduino has access to storage on an SD card and because of the process of server and client interaction.

The method that has been discussed can be applied to control Arduino I/O for various types of control logic involving logic elements, a timer, a counter, and a comparator written in JavaScript syntax expressions, even though the discussion here is focused on the Arduino Yun syntax.

4. Conclusion

The I/O of Arduino Yun was controlled through Boolean JavaScript expressions, which were programmed through a web-based UI. The control logic involved includes timer, counter, and comparator functions. The control logic can be changed during the run time and when controlling the Arduino working as a standalone unit. Digital input and output signals and the timer setting duration are delayed 0.3 s. Control signals involving analog signals require a cycle time of no longer than 0.53 s.

Acknowledgments

The authors thank the Electronics Cooling and Fuel Cell Laboratory of the Department of Refrigeration, Air-Conditioning and Energy Engineering of National Chin-Yi University of Technology, Taiwan (R.O.C.), and the Electrical and Instrument Laboratory of the Department of Refrigeration and Air Conditioning Engineering of Politeknik Negeri Bandung, Indonesia, for support in the system design and experimental activities.

References

1 arduino.cc: https://www.arduino.cc (accessed 8 April 2020).

2 M. El-Abd: Int. J. Eng. Pedagogy (iJEP) **7** (2017) 79. https://doi.org/10.3991/ijep.v7i2.6845

3 S. W. Kim and Y. Lee: J. Theor. Appl. Inf. Technol. (JATIT) **95** (2017) 4367. http://www.jatit.org/volumes/Vol95No18/4Vol95No18.pdf

4 A. P. G. Luciano, P. A. Fusinato, L. C. Gomes, A. Luciano, and H. Takai: J. Phys.: Conf. Ser. **1286** (2017) 012044. https://doi.org/10.1088/1742-6596/1286/1/012044

5 A. Mir and R. Swarnalatha: J. Eng. Sci. Technol. **13** (2018) 4131. http://jestec.taylors.edu.my/Vol%2013%20issue%2012%20December%202018/13_12_18.pdf

6 A. Singh, A. A. Kirubaraj, S. Senith, and S. R. J. Ramson: Int. J. Innov. Technol. Explor. Eng. (IJITEE) **8** (2018) 776. https://www.ijitee.org/wp-content/uploads/papers/v8i6/F3675048619.pdf

7 C. Rajan, B. Megala, A. Nandhini, and C. R. Priya: Int. J. Mech. Mater. Eng. **9** (2015) 365. https://doi.org/10.5281/zenodo.1100386

8 E. S. Maarif and T. Moyo: IOP Conf. Ser.: Mater. Sci. Eng. **535** (2019) 012016. https//doi.org/ 10.1088/1757-899X/535/1/012016

9 L. Louis: Int. J. Control Autom. Commun. Syst. (IJCACS) **1** (2016) 21. https://doi.org/10.5121/ijcacs.2016.1203

10 Parallax Inc.: https://www.parallax.com/downloads/plx-daq (accessed 8 April 2020).

11 Writing a Library for Arduino: https://www.arduino.cc/en/Hacking/libraryTutorial (accessed 8 April 2020).

12 Arduino Support from MATLAB: https://www.mathworks.com/hardware-support/arduino-matlab.html (accessed 8 April 2020).

13 LDmicro: https://github.com/LDmicro/LDmicro/releases (accessed 8 April 2020).

14 Windows Virtual Shields for Arduino: https://www.microsoft.com/en-us/p/windows-virtual-shields-for-arduino/9nblgggz0mld?activetab=pivot:overviewtab (accessed 8 April 2020).

15 G. Mahalakshmi and M. Vigneshwaran: Int. J. Eng. Adv. Res. Technol. (IJEART) **3** (2017) 7. https://www.ijeart.com/download_data/IJEART03803.pdf

16 S. S. H. Hajjaj, M. T. H. Sultan, M. H. Moktar, and S. H. Lee: Water **12** (2020) 502. https://doi.org/10.3390/w12020502

17 J. R. C. León, R. F. Martínez-González, A. M. Medinay, and L. A. Peralta-Pelaez: Int. J. Comput. Sci. Inf. Technol. **9** (2017) 97. https://doi.org/10.5121/ijcsit.2017.9508

18 D. Praveena, P. Keerthipriya, S. Vaijynthi, and D. Arvind: Int. J. Comput. Sci. Mobile Comput. (IJCSMC) **9** (2020) 199. https://ijcsmc.com/docs/papers/March2020/V9I3202054.pdf

19 Proportional on Measurement – The Code: http://brettbeauregard.com/blog/ (accessed 8 April 2020).

20 eFLL: https://github.com/zerokol/eFLL (accessed 8 April 2020).

21 Getting Started with the Arduino Yún: https://www.arduino.cc/en/Guide/ArduinoYun (accessed 8 April 2020).

22 Johnny-Five: http://johnny-five.io/ (accessed 8 April 2020).

23 G. Lockridge, B. Dzwonkowski, R, Nelson, and S. Powers: Sensors **16** (2016) 528. https://doi.org/10.3390/s16040528

24 C. T. Chao, M. H. Chung, J. S. Chiou, and C. J. Wang: Sensors **16** (2016) 435. https://doi.org/10.3390/s16040435

25 S. Lad, B. Mahajan, R. Mandkulkar, V. Pugaonkar, and K. Sailakshmi: Int. Res. J. Eng. Technol. (IRJET) **4** (2017) 1852. https://www.irjet.net/archives/V4/i4/IRJET-V4I4387.pdf

26 J. Simões, E. A. R. De Seabra, and A. Fernandes: 21st Int. Congr. Project Management and Engineering (2017) 1895–1904.

27 I. G. Pérez, A. J. C. Godoy, and M. C. Godoy: Proc.16th Int. Conf. Informatics in Control, Automation and Robotics (ICINCO) (2019) 174–180.

About the Authors

Pratikto received his B.S., M.S., and Ph.D. degrees from Institut Teknologi Bandung, Indonesia, in 1992, 2004, and 2010, respectively. Since 2012, he has been an associate professor at Politeknik Negeri Bandung, Indonesia. His research interests are in automation, software application, and sensors. (pratikto@polban.ac.id)

Raydha Zul Fitriani received her B.S. degree from Politeknik Negeri Bandung, Indonesia, in 2019 and is currently pursuing her M.S. at the Department of Refrigeration, Air Conditioning and Energy Engineering, National Chin-Yi University of Technology (NCUT), Taiwan. Her current research interests are in fuel cells. (s4A816028@student.ncut.edu.tw)

Yean-Der Kuan is a distinguished professor and former chairman (2013/02–2019/01) of the Department of Refrigeration, Air Conditioning and Energy Engineering, National Chin-Yi University of Technology, Taichung City, Taiwan. He received his Ph.D. degree from the Department of Mechanical and Aerospace Engineering at the University of Missouri, USA, in 2000. Currently, he is the director of the Taiwan Society of Heating, Refrigeration and Air Conditioning, the director of the Taiwan Energy Association, the director of the Taiwan Association for Hydrogen Energy and Fuel Cells, and a member of the American Society of Heating, Refrigerating, and Air-Conditioning. His research interests include the fields of energy saving and renewable energies, and air-conditioning components and systems.
(ydkuan@ncut.edu.tw)