



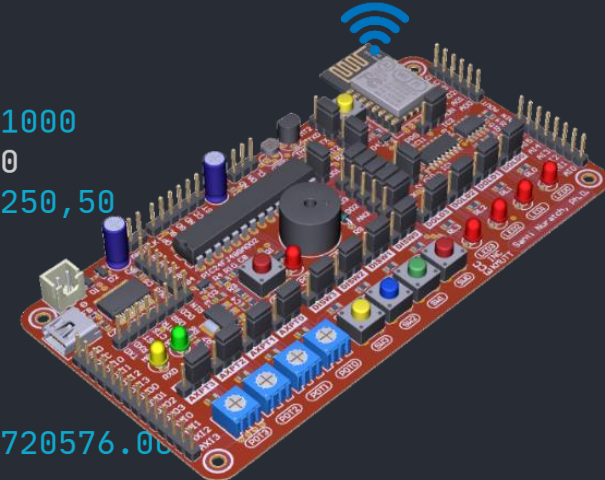
- If you can make communication through the serial port, you can control and monitor all peripherals of the board.
- If you can make communication through the internet, you can control and monitor all peripherals of the board anywhere around the world.

This document is written for general users who desire to control the microcontroller board, provided by the ECC-Lab, using text-based commands.

If you are an Embedded C programmer or C developer for embedded systems, please refer to the "ECC-RTOS" document.

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
blk,2,0,100
ok: led,2,0,100
beep,100
err: beep,100
beep,100,1000
err: beep,100,1000
buz,100,1250,50
ok: beep,100,1250,50
psw,1
ok: psw,1,0
adc,2
ok: adc,2,102
clk,0
ok: time,0,175720576.000
clk,1
ok: time,1,178780.766
clk,2
ok: time,2,0:3:0 500
```

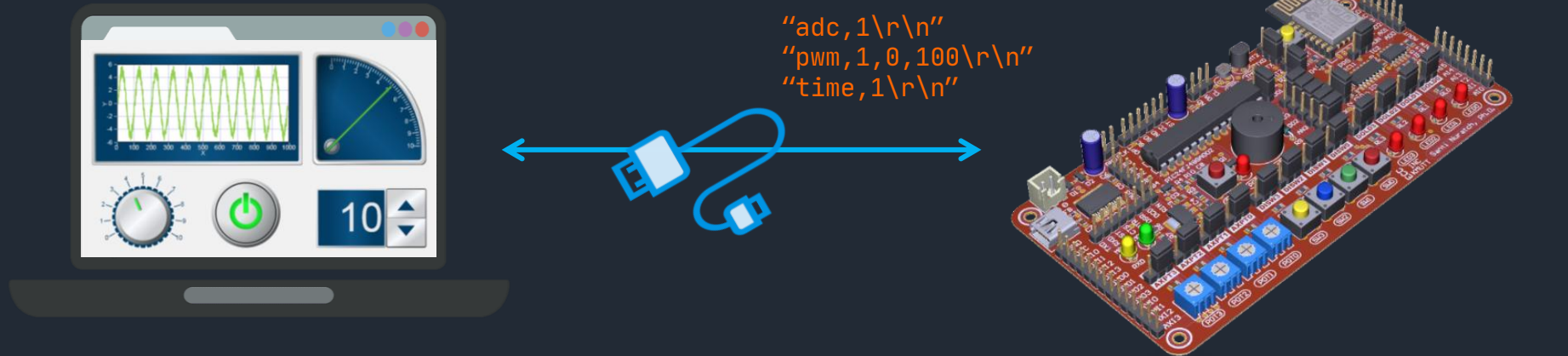


What is the ECC-Cmdex?

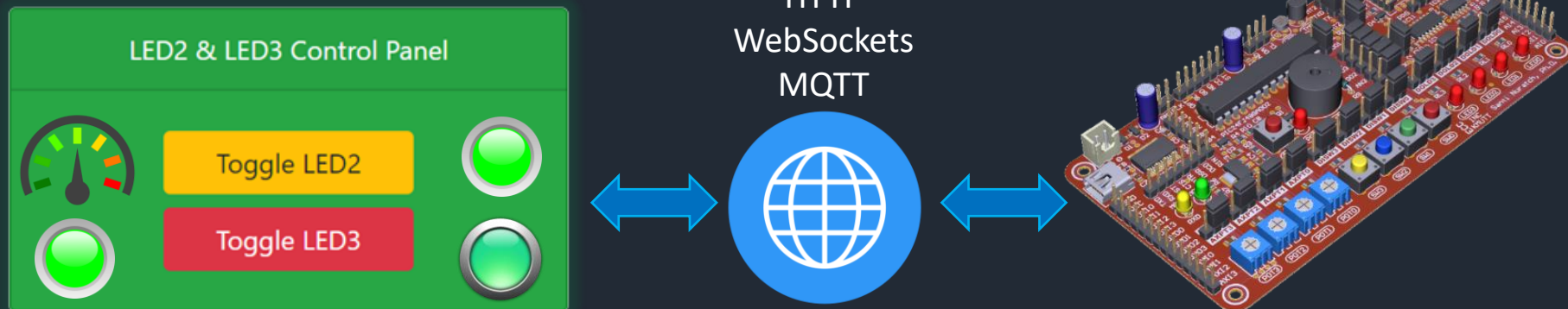


- The ECC-Cmdex is a special firmware developed for remote control and monitoring applications.
- It is a special embedded text processor, receiving, processing and executing the command.
- The firmware can receive the text-based commands through the serial port, WiFi and another communication channels, e.g., SPI and I2C.

Industrial Application Example



IoT Application Example





Commands (for general control and monitoring applications) (support both Serial-port-based and WiFi-based applications)

CMD	Full Name	Description
led	LED	Basic LED (Do) control command
fls	Flash	LED (Do) Flashing command
blk	Blink	LED (Do) Blinking command
cps	Continuous Pulse Signal	LED (Do) Pulse-Width modulation output command
buz	Buzzer	Buzzer/Speaker sound generating command
pwm	Pulse Width Modulation	High-Frequency PWM signal generation command
psw	Push Button Switch	PSW (Di) reading command
adc	Analog-to-Digital	ADC (Ai) reading command
det	ADC Detector	ADC (Ai) changing detector setting command
clk	Clock or Time	System Click and System Time reading command
jap	Join Access Point	Make the WiFi module to connect to the AP
rst	Restart MCU and WiFi	MCU and WiFi Restarting command

Note: The **jap** and **rst** are not allowed in this version, for safety reasons.



CMD:

```
led,id,functionCRLF
```

id:

```
0 - LED0 (DX00)
1 - LED1 (DX01)
2 - LED2 (DX02)
3 - LED3 (DX03)
```

function:

```
0 - clear
1 - set
2 - toggle
3 - read
```

Change status of LED0 to OFF

```
led,0,0CRLF
```

Change status of LED1 to ON

```
led,1,1CRLF
```

Changed status of LED2 to opposite status

```
led,2,2CRLF
```

Read status of the LED3

```
led,3,3CRLF
```

RET: (ok)

```
ok: led,id,D CRLF
```

D is the status of the target LED after the command is executed:

```
0 - OFF
1 - ON
```

RET: (error)

```
err: <wrong command>CRLF
```

```
ok: led,0,0CRLF
```

```
ok: led,1,1CRLF
```

D is the status of the LED2, 0 is OFF, 1 is ON

```
ok: led,2,D CRLF
```

D is the status of the LED3, 0 is OFF, 1 is ON

```
ok: led,3,D CRLF
```



CMD:

```
fls,id,intervalCRLF
```

id:

- 0 - LED0 (DX00)
- 1 - LED1 (DX01)
- 2 - LED2 (DX02)
- 3 - LED3 (DX03)

interval:

1 to 65535
(milliseconds)

Flash the LED0 for 10 mS

```
fls,0,10CRLF
```

Flash the LED1 for 123 mS

```
fls,1,125CRLF
```

Flash the LED2 for 1 Sec

```
fls,2,1000CRLF
```

Flash the LED3 for 5.5 Sec

```
fls,3,5500CRLF
```

RET: (ok)

```
ok: fls,id,intervalCRLF
```

RET: (error)

```
err: <wrong command>CRLF
```

```
ok: led,0,10CRLF
```

```
ok: led,1,125CRLF
```

```
ok: led,2,1000CRLF
```

```
ok: led,3,5500CRLF
```



CMD:

```
blk,id,delay,intervalCRLF
```

id: interval:
0 - LED0 (DX00) 1 to 65535
1 - LED1 (DX01) (milliseconds)
2 - LED2 (DX02)
3 - LED3 (DX04)

delay:
1 to 65535
(milliseconds)

Blink the LED0 with 10 mS delay and 20 mS interval

```
blk,0,10,20CRLE
```

Blink the LED0 with 50 mS delay and 100 mS interval

```
blk,1,50,100CRLE
```

Blink the LED0 with 1 Sec delay and 50 mS interval

```
blk,2,1000,50CRLE
```

Blink the LED0 with 500 mS delay and 2 Sec interval

```
blk,3,500,2000CRLE
```

RET: (ok)

```
ok: blk,id,delay,intervalCRLF
```

RET: (error)

```
err: <wrong command>CRLF
```

```
ok: blk,0,10,20CRLE
```

```
ok: blk,1,50,100CRLE
```

```
ok: blk,2,1000,50CRLE
```

```
ok: blk,3,500,2000CRLE
```



CMD:

```
cps,id,delay,width,periodCRLF
```

id:	width:
0 - LED0 (DX00)	1 to 65535
1 - LED1 (DX00)	(milliseconds)
2 - LED2 (DX00)	
3 - LED3 (DX00)	

delay:	period:
1 to 65535	1 to 65535
(milliseconds)	(milliseconds)

Generate continuous pulse for LED0 with
10 mS delay, 20 mS width, 100 mS period

```
cps,0,10,20,20CRLF
```

Generate continuous pulse for LED1 with
0 mS delay, 10 mS width, 500 mS period

```
cps,1,0,10,500CRLF
```

Generate continuous pulse for LED2 with
100 mS delay, 50 mS width, 5 Sec period

```
cps,2,100,50,5000CRLF
```

Generate continuous pulse for LED3 with
500 mS delay, 2 Sec width, 3.5 Sec period

```
cps,3,500,2000,3500CRLF
```

RET: (ok)

```
ok: cps,id,delay,width,periodCRLF
```

RET: (error)

```
err: <wrong command>CRLF
```

```
ok: cps,0,10,20,20CRLF
```

```
ok: cps,1,0,10,500CRLF
```

```
ok: cps,2,100,50,5000CRLF
```

```
ok: cps,3,500,2000,3500CRLF
```



CMD:

```
buz, interval, freq, powerCRLF
```

interval: freq: power:
1 to 65535 1 to 65535 0 to 100
(milliseconds) (Hertz) (percent)

```
buz, interval, freqCRLF
```

```
buz, intervalCRLF
```

Generate beep sound for 200 mS, 1 kHz, 50% power

```
buz, 200, 1000, 50CRLF
```

Generate beep sound for 500 mS, 3.5 kHz
(previous power is used)

```
buz, 500, 3500CRLF
```

Generate beep sound for 1000 mS, 3.5 kHz
(previous frequency and power are used)

```
buz, 1000CRLF
```

Generate beep sound for 500 mS, 5.5 kHz, 90% power

```
buz, 500, 5500, 90CRLF
```

RET: (ok)

```
ok: buz, interval, freq, powerCRLF
```

RET: (error)

```
err: <wrong command>CRLF
```

```
ok: buz, 200, 1000, 50CRLF
```

```
ok: buz, 500, 3500CRLF
```

```
ok: buz, 1000CRLF
```

```
ok: buz, 500, 5500, 90CRLF
```




CMD:

```
pwm,id,function,valueCRLF
```

id:	function:
0 - PWM0 (SCL)	0 - set frequency
1 - PWM1 (SDA)	1 - set duty cycle ratio
2 - PWM2 (LED2)	2 - set phase-shift ratio
3 - PWM3 (LED3)	3 - stop/start (0/1)

value:
positive real or integer number
depended on the function.

frequency: 0.95Hz-160kHz, ratio: 0.0-1.0

Set frequency of PWM0 to 2 kHz

```
pwm,0,0,2000CRLF
```

Set duty cycle ratio of PWM1 to 0.2

```
pwm,1,1,0.2CRLF
```

Set phase-shift ratio of PWM2 to 0.5

```
pwm,2,2,0.5CRLF
```

Stop PWM3

```
pwm,3,3,0CRLF
```

Start PWM3

```
pwm,3,3,1CRLF
```

RET: (ok)

```
ok: pwm,id,function,valueCRLF
```

RET: (error)

```
err: <wrong command>CRLF
```

```
ok: pwm,0,0,2000CRLF
```

```
ok: pwm,1,1,0.2CRLF
```

```
ok: pwm,2,2,0.5CRLF
```

```
ok: pwm,3,3,0CRLF
```

```
ok: pwm,3,3,1CRLF
```



CMD:

```
psw,idCRLF
```

id:

- 0 - PSW0 (DXI0)
- 1 - PSW1 (DXI1)
- 2 - PSW2 (DXI2)
- 3 - PSW3 (DXI3)

RET: (ok)

```
ok: psw,id,D CRLF
```

D is the status of the target switch:

- 0 - OFF
- 1 - ON

RET: (error)

```
err: <wrong command>CRLF
```

Get status of the PSW0

```
psw,0CRLF
```

Get status of the PSW1

```
led,1CRLF
```

Get status of the PSW2

```
led,2CRLF
```

Get status of the PSW3

```
led,3CRLF
```

PSW0 is OFF

```
ok: psw,0,0CRLF
```

PSW0 is ON

```
ok: psw,0,1CRLF
```

PSW1 is OFF

```
ok: psw,1,0CRLF
```

PSW1 is ON

```
ok: psw,1,1CRLF
```

PSW2 is OFF

```
ok: psw,2,0CRLF
```

PSW2 is ON

```
ok: psw,2,1CRLF
```

PSW3 is OFF

```
ok: psw,3,0CRLF
```

PSW3 is ON

```
ok: psw,3,1CRLF
```



CMD:

```
adc, idCRLF
```

id:

- 0 - ADC0 (POT0/AXI0)
- 1 - ADC1 (POT1/AXI1/LDR)
- 2 - ADC2 (POT2/AXI2)
- 3 - ADC3 (POT3/AXI3)

Get status of the ADC0

```
adc, 0CRLF
```

Get status of the ADC1

```
adc, 1CRLF
```

Get status of the ADC2

```
adc, 2CRLF
```

Get status of the ADC3

```
adc, 3CRLF
```

RET: (ok)

```
ok: adc, id, D CRLF
```

D is the 10-bit data of the target ADC
(0x000 to 0x3FF or 0 to 1023)

RET: (error)

```
err: <wrong command>CRLF
```

ADC0 value is 64

```
ok: psw, 0, 64CRLF
```

ADC0 value is 128

```
ok: psw, 1, 128CRLF
```

ADC0 value is 256

```
ok: psw, 2, 256CRLF
```

ADC0 value is 512

```
ok: psw, 3, 512CRLF
```



CMD:

```
clk,typeCRLF
```

type:

- 0 - HH:MM:SS:xxx
- 1 - Microseconds
- 2 - Milliseconds

Get system time in format of HH:MM:SS:xxx

```
clk,0CRLF
```

Get system time in format of microseconds

```
clk,1CRLF
```

Get system time in format of milliseconds

```
clk,2CRLF
```

RET: (ok)

```
ok: ckk,type,D CRLF
```

D is the system time information

RET: (error)

```
err: <wrong command>CRLF
```

```
ok: time,0,hh:mm:ss:xxx CRLF
```

```
ok: time,1,xxx.xxx CRLF
```

```
ok: time,2,xxx.xxx CRLF
```



CMD:

```
det,id,threshold,intervalCRLF
```

id:	threshold:
0 - ADC0 (POT0/AXI0)	0 to 1023
1 - ADC1 (POT1/AXI1/LDR)	
2 - ADC2 (POT2/AXI2)	interval:
3 - ADC3 (POT3/AXI3)	0 to 65535

Set auto-detected threshold of the ADC0 to 50

```
det,0,50CRLF
```

Set auto-detected threshold and interval of the ADC0 to 50 and 500 respectively

```
det,0,50,500CRLF
```

Set auto-detected threshold of the ADC3 to 100

```
det,3,100CRLF
```

Set auto-detected threshold and interval of the ADC3 to 100 and 2000 respectively

```
det,3,100,2000CRLF
```

RET: (ok)

```
ok: det,id,threshold,intervalCRLF
```

RET: (error)

```
err: <wrong command>CRLF
```

```
ok: det,0,50CRLF
```

```
ok: det,0,50,500CRLF
```

```
ok: det,3,100CRLF
```

```
ok: det,3,100,2000CRLF
```



CMD:

```
jap,ssid,pass,CRLF
```

ssid:

The Service Set Identifier

pass:

The password of Wi-Fi network

Connect to the network with the given ssid and pass

```
jap,ecc-lab,@eccpasswCRLF
```

RET: (ok)

```
ok: jap,ssidCRLF
```

RET: (error)

```
err: <wrong command>CRLF
```

```
ok: jap,ecc-labCRLF
```

Note: The **jsp** is not allowed in this version, for safety reasons.



CMD:

```
rst,idCRLF
```

id:

```
0 - MCU  
1 - WiFi
```

RET: (ok)

```
ok: rst,idCRLF
```

RET: (error)

```
err: <wrong command>CRLF
```

Restart the MCU

```
rst,0CRLF
```

NO RETURN

Restart the WiFi

```
rst,1CRLF
```

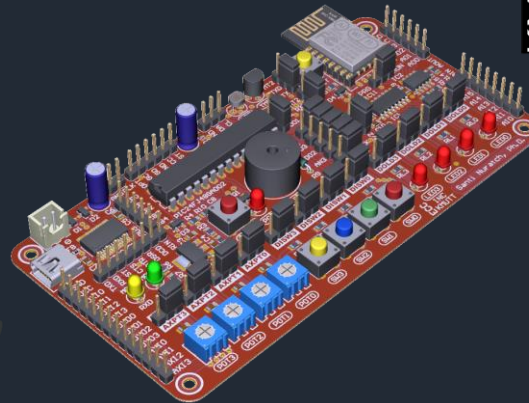
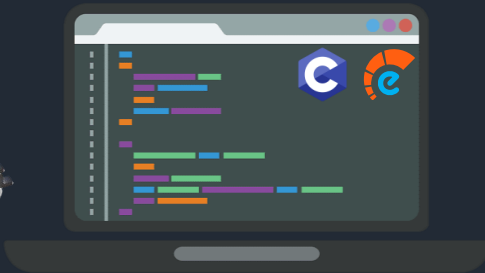
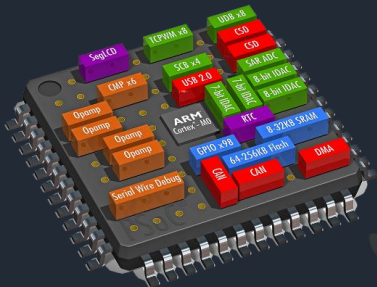
```
ok: rst,1CRLF
```

Note: The `rst` is not allowed in this version, for safety reasons.

THANK YOU!



We Make Computers do More



Asst.Prof.Dr.Santi Nuratch

Embedded Computing and Control Lab. @ INC-KMUTT

santi.inc.kmutt@gmail.com

Department of Control System and Instrumentation Engineering,
King Mongkut's University of Technology Thonburi, **KMUTT**