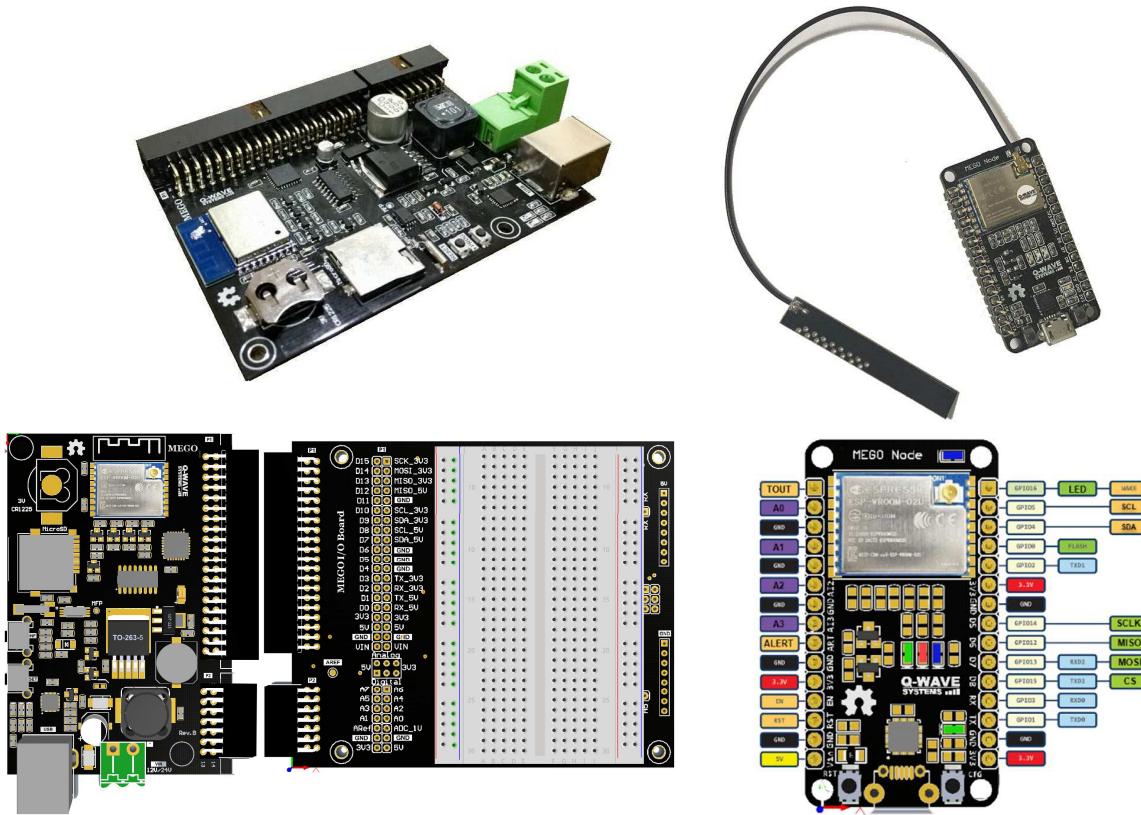


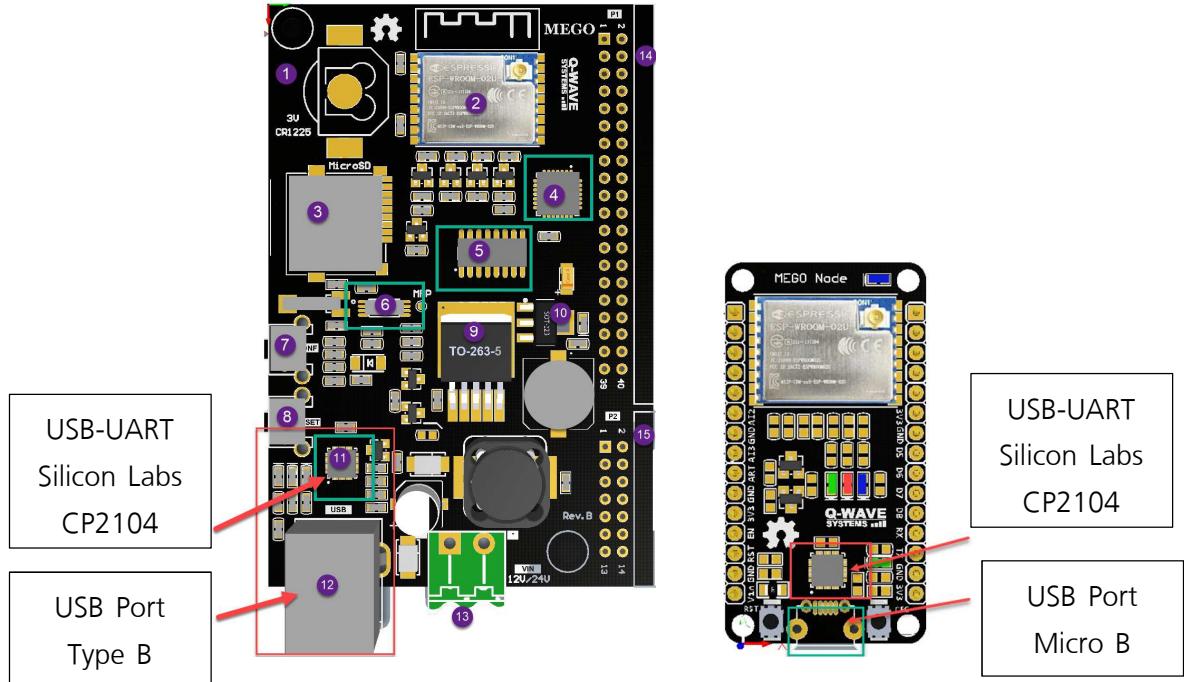
## การใช้งานบอร์ด MEGO และ MEGO Node เปื้องตัน



บอร์ด MEGO (ซ้าย) และบอร์ด MEGO Node (ขวา)

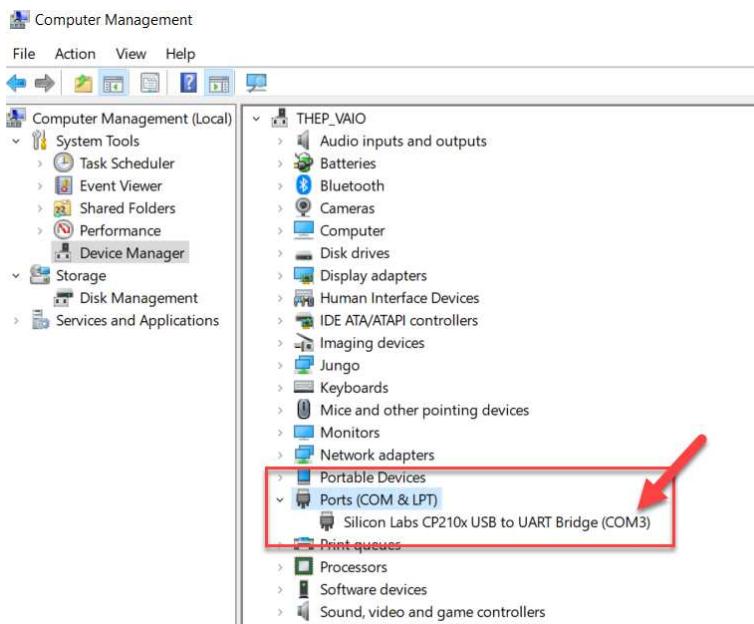
### การติดตั้ง MEGO USB Driver

บอร์ด MEGO จะมีวงจรในส่วนของ USB-UART Bridge เพื่อเชื่อมต่อกับคอมพิวเตอร์ผ่านทางพอร์ต USB โดยจะคอมพิวเตอร์จะมองเห็นบอร์ดเชื่อมต่อเป็นแบบ COM Port อาทิ "COM3" ซึ่งเอาไว้สำหรับดาวน์โหลดโปรแกรมลงในชิปในครอคูนทอร์เลอร์ MCU (WiFi-Soc) และเอาไว้สื่อสารกับไมโครครูนทอร์เลอร์ผ่านทางพอร์ต UART สำหรับดีบักโปรแกรม หรือรับส่งข้อมูลกับคอมพิวเตอร์

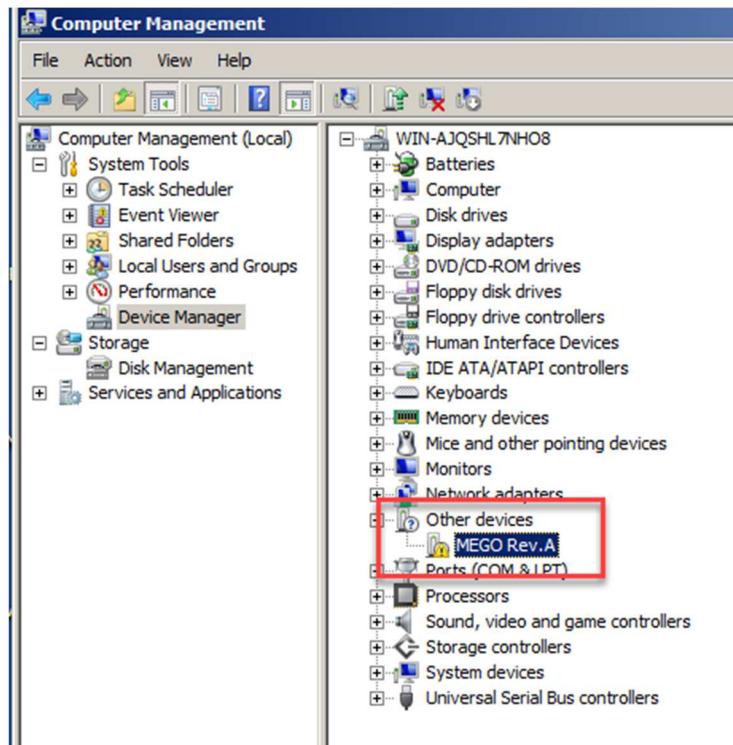


บอร์ด MEGO Development Kit และบอร์ด MEGO Node

เมื่อเสียบบอร์ดเข้ากับคอมพิวเตอร์ ให้ตรวจสอบที่ Device Manager ถ้ามองเห็นบอร์ดซึ่งมีต่อ COM Port เป็น COM Port อาทิ “COM3” แสดงว่าพร้อมใช้งาน ปกติดัก้าใช้ Windows 10 จะมองเห็นบอร์ดทันที



แต่ถ้าใช้งาน Windows เวอร์ชันเก่า อาทิ Windows 7 หรือ Windows 8.1 นักจะมองไม่เห็น บอร์ด ต้องติดตั้ง Driver ก่อนใช้งาน



ในกรณีที่ไม่พบ COM Port ให้ทำการดาวน์โหลด “CP210x USB to UART Bridge VCP Drivers” ตามลิงค์ด้านล่าง โดยให้ดาวน์โหลดเวอร์ชันตาม Windows ที่ใช้งาน แสดงดังรูป

<https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

Download for Windows 10 Universal (v10.1.1)

Platform	Software	Release Notes
Windows 10 Universal	<a href="#">Download VCP (2.3 MB)</a>	<a href="#">Download VCP Revision History</a>

Download for Windows 7/8/8.1/10 (v6.7.5)

Platform	Software	Release Notes
Windows 7/8/8.1/10	<a href="#">Download VCP (5.3 MB) (Default)</a>	<a href="#">Download VCP Revision History</a>
Windows 7/8/8.1/10	<a href="#">Download VCP with Serial Enumeration (5.3 MB)</a>	<a href="#">Download VCP Revision History</a>

วิธีการเลือกหนึ่งในการสามารถดาวน์โหลดได้จาก Github Server ของทาง Q-Wave Systems “<https://github.com/QWaveSystems>”

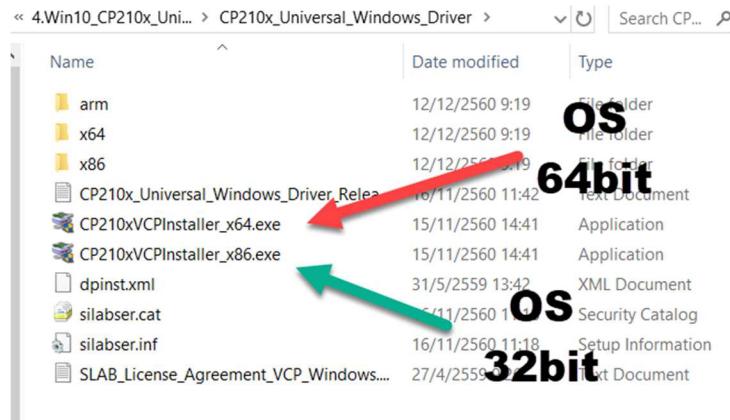
โดยสำหรับผู้ที่ใช้ Windows 7 หรือ Windows 8.1 ดาวน์โหลดที่

[https://github.com/QWaveSystems/MEGO/blob/master/MEGO-USB\\_Driver/4.Win7\\_8\\_Cp210x\\_Windows\\_Drivers.zip](https://github.com/QWaveSystems/MEGO/blob/master/MEGO-USB_Driver/4.Win7_8_Cp210x_Windows_Drivers.zip)

โดยสำหรับผู้ที่ใช้ Windows 10 ดาวน์โหลดได้ที่

[https://github.com/QWaveSystems/MEGO/blob/master/MEGO-USB\\_Driver/4.Win10\\_CP210x\\_Universal\\_Windows\\_Driver.zip](https://github.com/QWaveSystems/MEGO/blob/master/MEGO-USB_Driver/4.Win10_CP210x_Universal_Windows_Driver.zip)

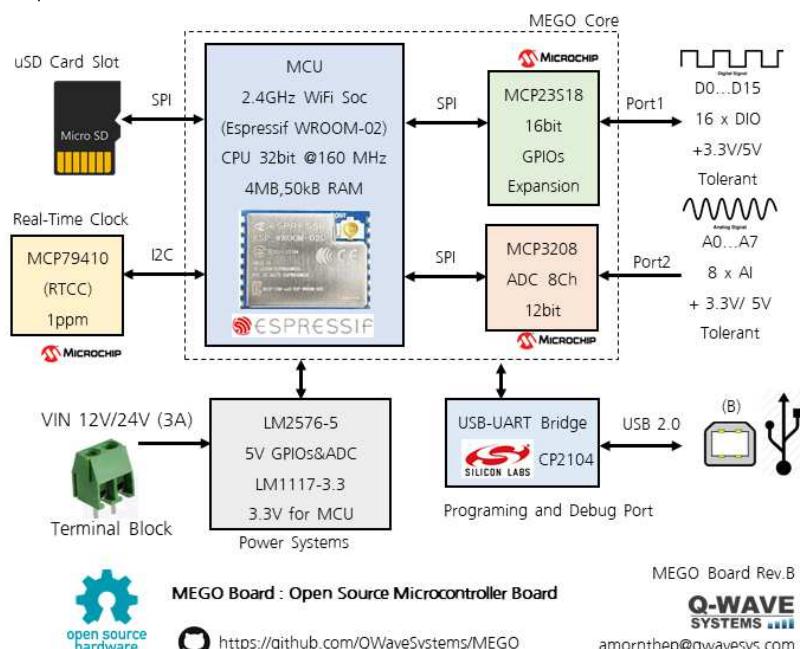
เมื่อดาวน์โหลดมาแล้วให้แตกไฟล์ Zip และติดตั้งไฟล์ตามระบบปฏิบัติการที่ใช้ โดยถ้าใช้ Windows 32 บิตให้ติดตั้งไฟล์ “\_x86.exe” และถ้าใช้งาน Windows 64 บิต ให้ติดตั้งไฟล์ “\_x64.exe”



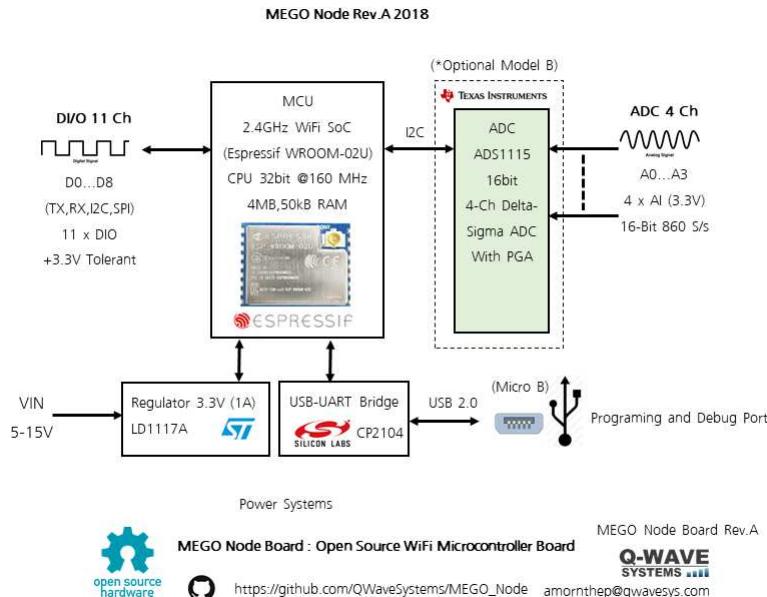
ทำความรู้จักบอร์ด MEGO เป็นต้น

บอร์ด MEGO (WiFi MCU Development Kit) และ MEGO Node เป็นบอร์ดพัฒนาระบบสมองกลฝังตัวเชื่อมต่อ WiFi (2.4GHz) โดยสมองกลของบอร์ดคือ ไมโครคอนโทรลเลอร์ (MCU) 32 bit ทำงานที่สัญญาณนาฬิกาความเร็ว 160 MHz เลือกใช้ MCU ยี่ห้อ Espressif WROOM-02 ที่ผ่านการรับรอง FCC,CE

เป็นบอร์ดที่มีประสิทธิภาพสูง เหมาะสำหรับพัฒนาระบบสมองกลฝังตัวเพื่อใช้งานในสภาพแวดล้อมจริง โดยบอร์ดมีอุปกรณ์เชื่อมต่อที่จำเป็นครบครัน



## Block Diagram ของบอร์ด MEGO



Block Diagram ของบอร์ด MEGO Node

## รายละเอียดทางด้านเทคนิคของบอร์ด MEGO

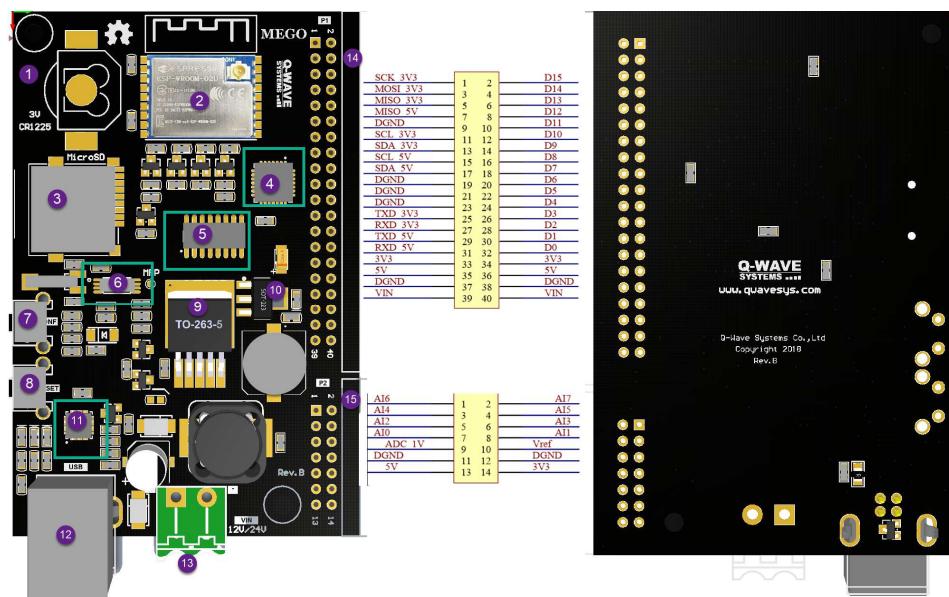
- ไมโครคอนโทรลเลอร์ MCU Espressif 32 bit @160 MHz (WROOM-02)
- รองรับการเชื่อมต่อผ่าน WiFi 2.4GHz , Memory Flash 2MB, RAM 50kB
- พอร์ตรับสัญญาณ Analog จำนวน 8 ช่อง (A0-A7), 12 bit ADC (3.3/5V)
- พอร์ตรับ-ส่งสัญญาณ Digital จำนวน 16 Digital (3.3V/5V) I/O (D0-D15) Supported (1-Wire,DHTxx,SoftSerial,SoftI2C)
- วงจรนาฬิกา RTC (Real-Time Clock) สามารถใช้กับนาฬิกา 3.3V (C)
- รองรับการเชื่อมต่อกับ Micro SD Card slot (uSD) รองรับสูงสุด 32GB
- ภาคจ่ายไฟ Power Supply Input 12V/24V จ่ายกระแสสูงสุด 3A
- รองรับการอัพโหลดโปรแกรมผ่าน USB debug and programming (Speed 921600)
- พอร์ตสื่อสารแบบ SPI = 1 Port (3.3V/5V)
- พอร์ตสื่อสารแบบ I2C = 1 Port (3.3/5V)
- พอร์ตสื่อสารแบบ Serial = 1 Port (3.3/5V)
- ขนาดของบอร์ด Dimension 85mm x56.2mm

## รายละเอียดทางด้านเทคนิคของบอร์ด MEGO Node

- ไมโครคอนโทรลเลอร์ MCU Espressif 32 bit @160 MHz (WROOM-02)
- รองรับการเชื่อมต่อผ่าน WiFi 2.4GHz , Memory Flash 2MB, RAM 50kB
- พอร์ตรับสัญญาณ Analog 4 ช่อง 16 bit (ADS1115) ADC (3.3V) \*Optional for Model B

- พортตระบ-ส่งสัญญาณ Digital I/O จำนวน 11 Digital (3.3V)
- รองรับการอัพโหลดโปรแกรมผ่าน USB debug and programming (Speed 921600)
- ภาคจ่ายไฟ Onboard Power regulator 3.3V 1A ,Supply Input VIN = 5V-15V Power +5V via USB port for programming and debugging
- พортสื่อสารแบบ SPI = 1 Port (3.3V) \*ใช้ร่วมกับขา Digital I/O
- พортสื่อสารแบบ I2C = 1 Port (3.3V) \*ใช้ร่วมกับขา Digital I/O
- พортสื่อสารแบบ Serial = 1 Port (3.3) \*ใช้ร่วมกับขา Digital I/O
- ขนาดของบอร์ด Dimension 49 x 24.5 x 13mm

ลักษณะของบอร์ด MEGO



รายการอุปกรณ์บนบอร์ด MEGO

- 1.Battery Holder CR1220/1225 (3.3V)
- 2.Espressif MCU WROOM-02
- 3.Micro-SD Card Slot
- 4.IC MCP23S18 (SPI) Expansion 16 bit GPIOs
- 5.IC MC3208 (SPI) ADC 12bit 8 Ch
- 6.IC RTCC Microchip MCP79410
- 7.User/Config Button
- 8.Reset Button
- 9.IC Switching Voltage Regulator LM2576-5 5V 3A
- 10.IC LDO Voltage Regulator LM1117 3.3V 1A
- 11.USB UART Bridge Silicon Labs CP2104
- 12.USB Type B Debug/Programming Port
- 13.Vin Connector DC 12V or 24V (Max 30V)

14.GPIO Port 1

15.GPIO Port 2

การจัดเรียงขา (Port 1)

Description	Pin	Pin	Description
(SPI) SCK 3.3V	1	2	D15
(SPI) MOSI 3.3V	3	4	D14
(SPI) MISO 3.3V	5	6	D13
(SPI) MISO 5V	7	8	D12
GND	9	10	D11
(I2C) SCL 3.3V	11	12	D10
(I2C) SDA 3.3V	13	14	D9
(I2C) SCL 5V	15	16	D8
(I2C) SDA 5V	17	18	D7
GND	19	20	D6
GND	21	22	D5
GND	23	24	D4
(Serial) TX 3.3V	25	26	D3
(Serial) RX 3.3V	27	28	D2
(Serial) TX 5V	29	30	D1
(Serial) RX 5V	31	32	D0
3.3V	33	34	3.3V
5V	35	36	5V
GND	37	38	GND
Vin 12V/24V	39	40	Vin 12V/24V

การจัดเรียงขา (Port 2)

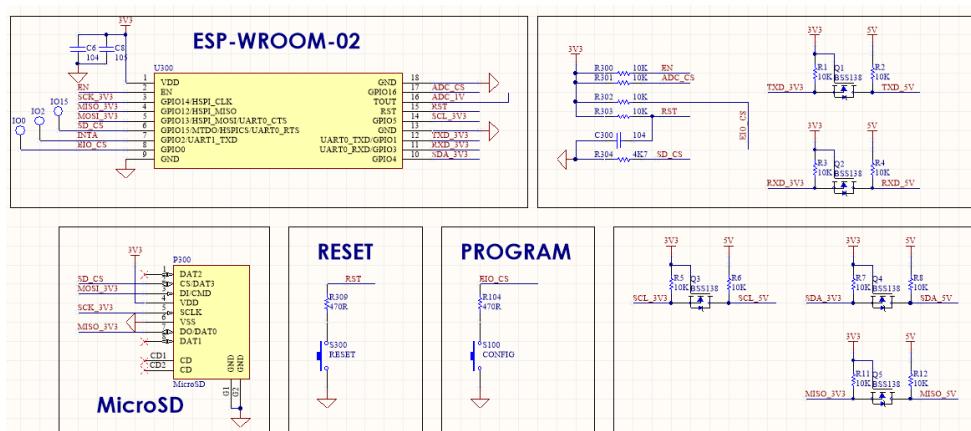
Description	Pin	Pin	Description
Analog A6	1	2	Analog A7
Analog A4	3	4	Analog A5
Analog A2	5	6	Analog A3
Analog A0	7	8	Analog A1

(ESP) ADC 1V	9	10	ADC VRef
GND	11	12	GND
5V	13	14	3.3V

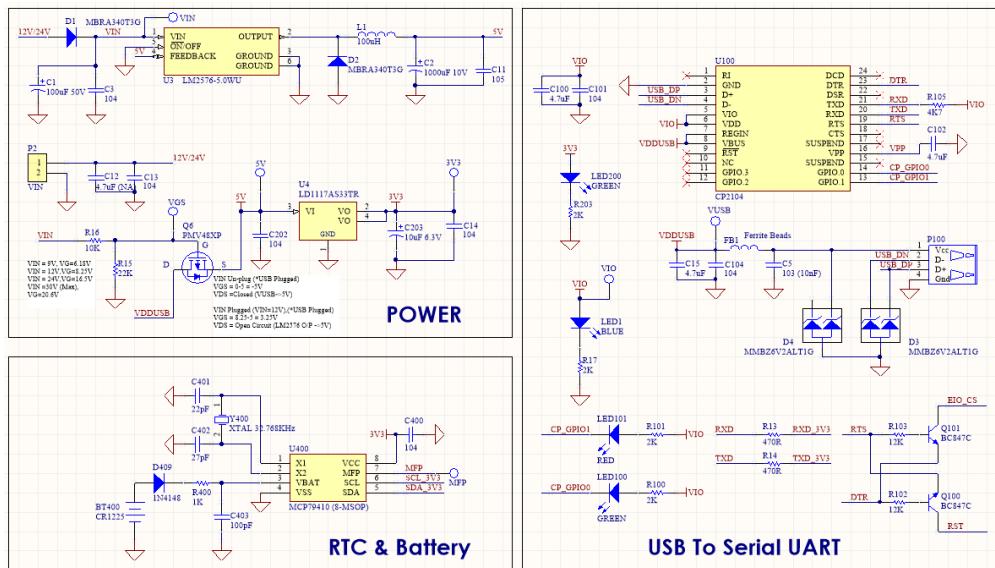
### การอ่านแบบวงจร MEGO (Schematic)

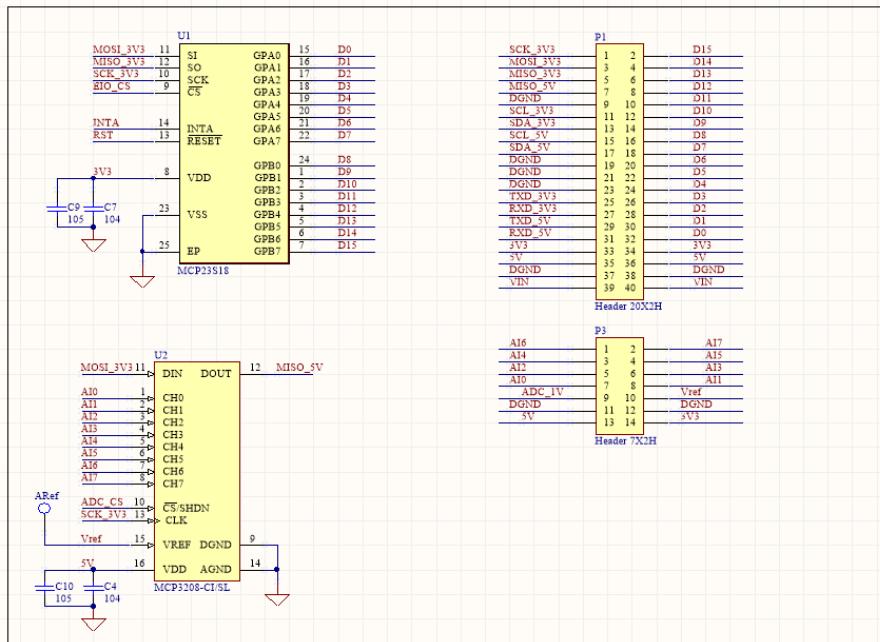
สามารถดาวน์โหลดไฟล์อ่านแบบได้ที่นี่

<https://github.com/QWaveSystems/MEGO>



### MEGO Rev.B



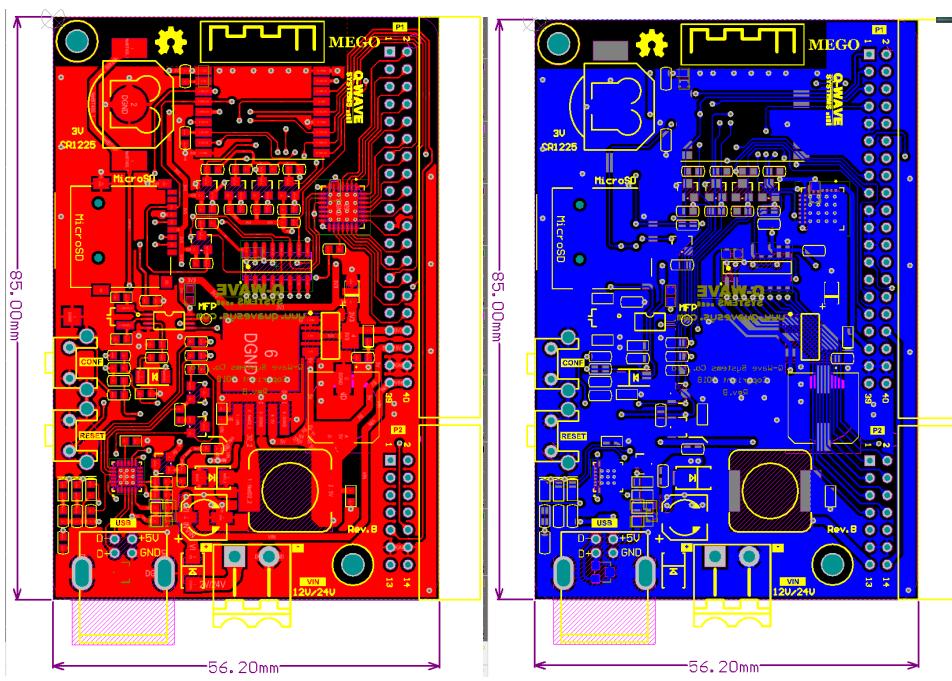


ภาพผัง Schematic ของบอร์ด MEGO Rev.B

### การจัดวาง Component และ PCB Layout

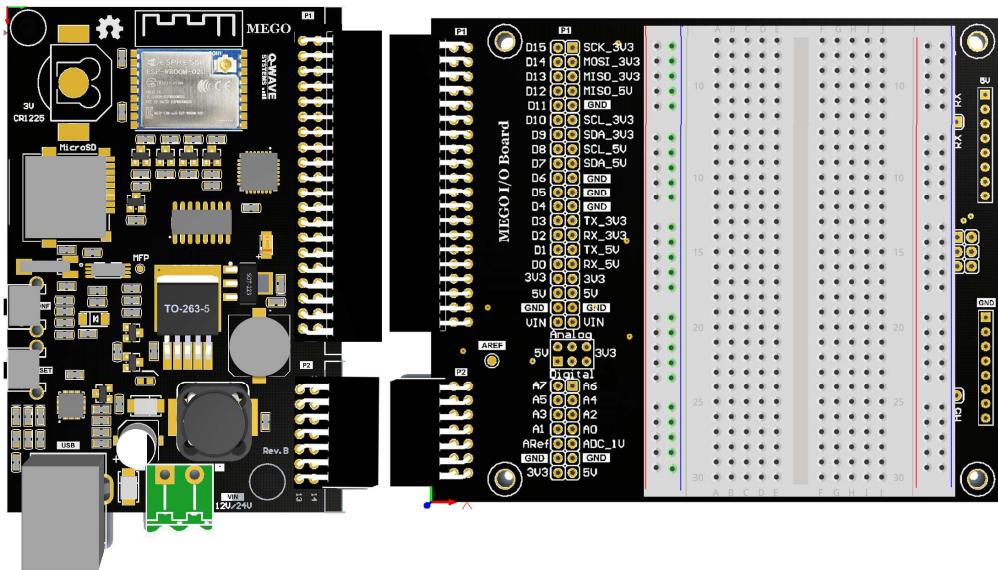
สามารถดาวน์โหลดไฟล์ออกแบบได้ที่นี่

<https://github.com/QWaveSystems/MEGO>

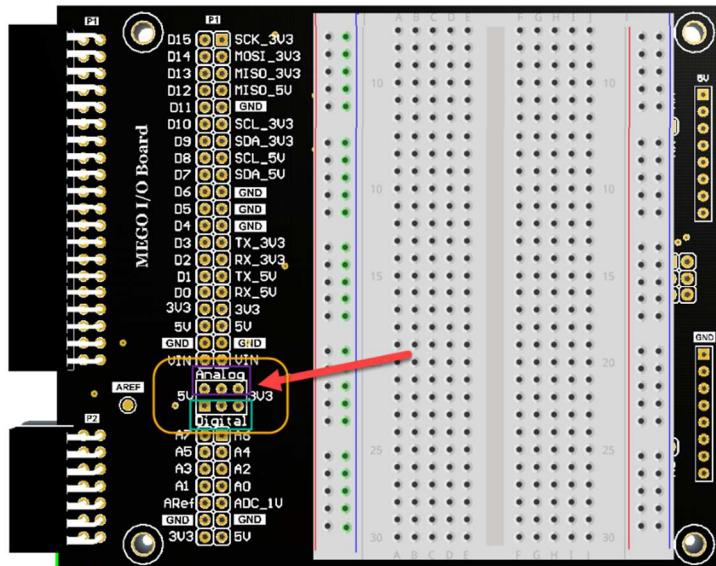


PCB Layout ของบอร์ด MEGO I/O

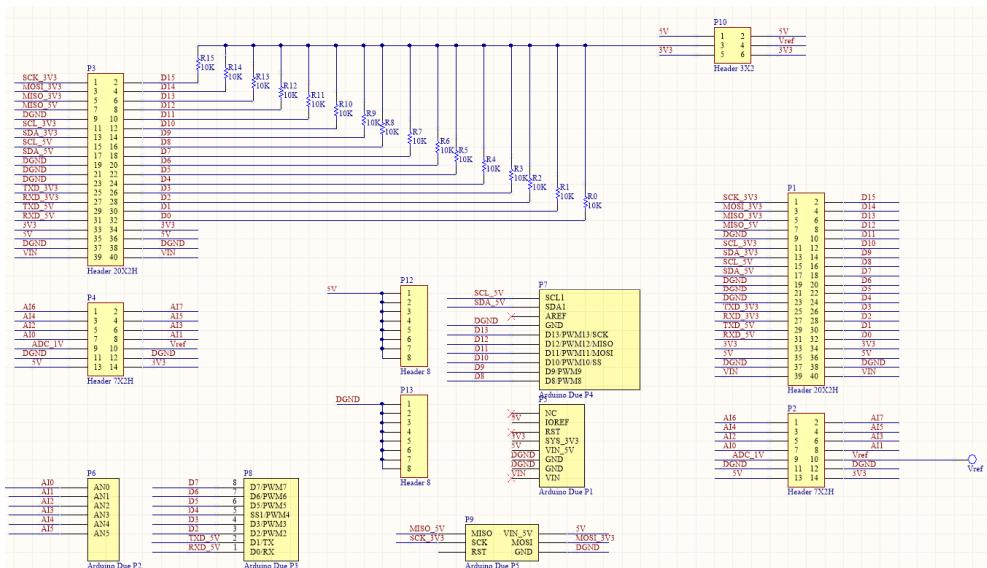
### การเชื่อมต่อ กับบอร์ด MEGO I/O และการกำหนด Jumper



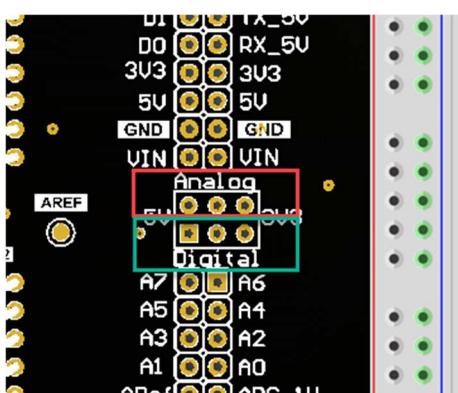
การเชื่อมต่อบอร์ด MEGO และบอร์ด MEGO I/O เข้าด้วยกัน



การกำหนด Jumper ของ Analog/Digital

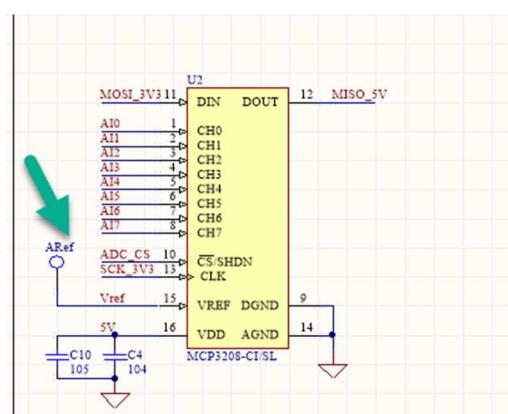


ແພນັກ Schematic ຂອງບ່ອർດ MEGO I/O Rev.B



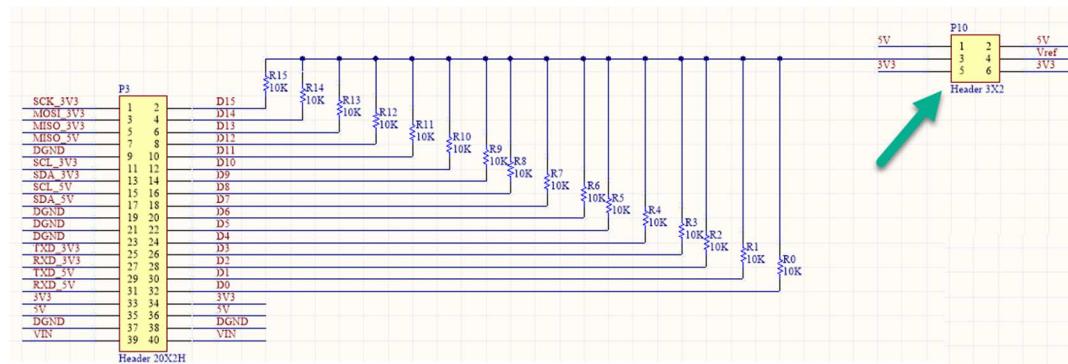
### Analog Jumper:

ບໍລິບ່ອຮດຈະໃຊ້ MCP3208 ADC 12bit ຈຳວນ 8 ຊ່ອງ ໂດຍສາມາດກຳໜົດສັງຄູານວ້າງວົງ  
Aref (Analog Reference) ກຳໜົດໂດຍ Jumper ດ້ວຍເຫັນ ໃນກຽນທີ່ເລືອກກາງດ້ານຊ້າຍ 5V ແລ້ວ  
"Analog Reference" ຈະວ້າງວົງກັບ 5V ໃຕ່ກ້າວເລື່ອນມາດ້ານຂວາຈະວ້າງວົງກັບ 3.3V

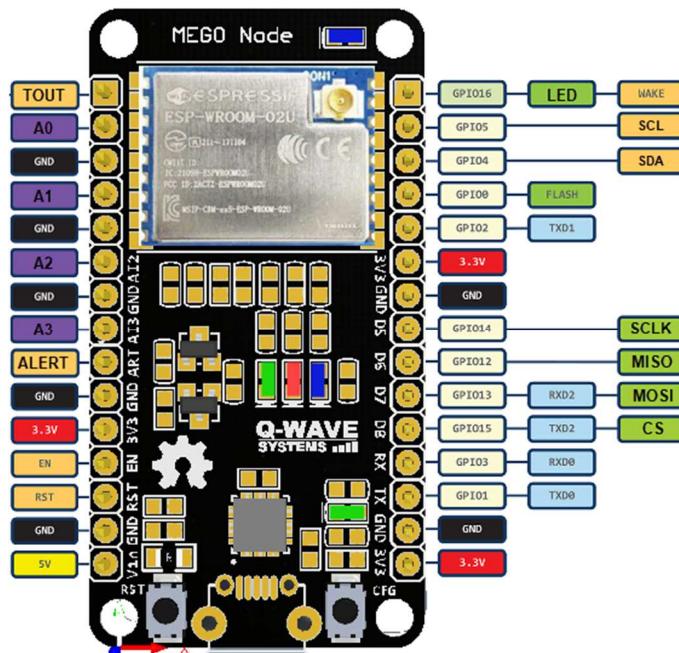


## Digital Jumper:

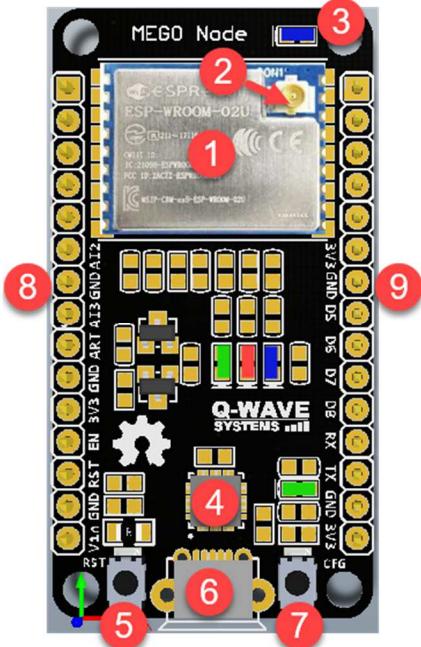
บอร์ด MEGO ใช้ชิป MCP23S18 ขยายพอร์ต Digital ให้มีจำนวน 16 DIO โดย DIO ทั้งหมดจะมีลักษณะเป็น “Open-Collector” โดยการใช้งานสามารถเลือกระดับสัญญาณอ้างอิงได้เอง โดยกำหนดโดย Jumper ด้านล่าง ในกรณีที่เลือกการด้านซ้าย 5V แสดงว่า “Digital Pull-Up” จะอ้างอิงกับ 5V แต่ถ้าเลื่อนมาด้านขวาจะอ้างอิงกับ 3.3V



ลักษณะของบอร์ด MEGO Node



รายการอุปกรณ์บนบอร์ด MEGO



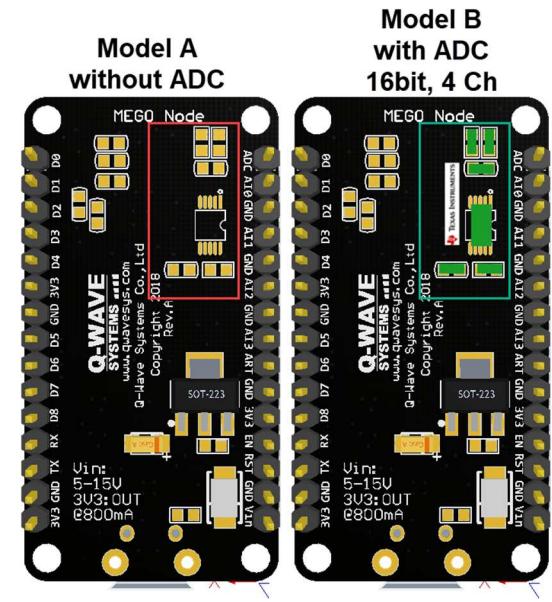
## Connector

	J1	J2
ADC EX	A0 1	PIN1 1 D0 GPIO16
AIN0	2	PIN2 2 D1 GPIO5
DGND	3	PIN3 3 D2 GPIO4
AIN1	4	PIN4 4 D3 GPIO0
DGND	5	PIN5 5 D4 GPIO2
AIN2	6	PIN6 6 3V3
DGND	7	PIN7 7 DGND
AIN3	8	PIN8 8 D5 GPIO14
ALERT	9	PIN9 9 D6 GPIO12
DGND	10	PIN10 10 D7 GPIO13
3V3	11	PIN11 11 D8 GPIO15
EN	12	PIN12 12 RX RXD
RST	13	PIN13 13 TX TXD
DGND	14	PIN14 14 DGND
VDD5V	15	PIN15 15 3V3

THT\_Male\_P\_1x15      THT\_Male\_P\_1x15

- 1.Espressif MCU WROOM-02
- 2.Antenna WiFi Connector U.FL 2.4GHz
- 3.LED Pin16 (GPIO)
- 4.USU UART Bridge Silicon Labs CP2104
- 5.Reset Button Switch (SW1)
- 6.Micro USB Connector (Programming&Debug)
- 7.User/Config Button (SW2)
- 8.GPIO Port 1
- 9.GPIO Port 2

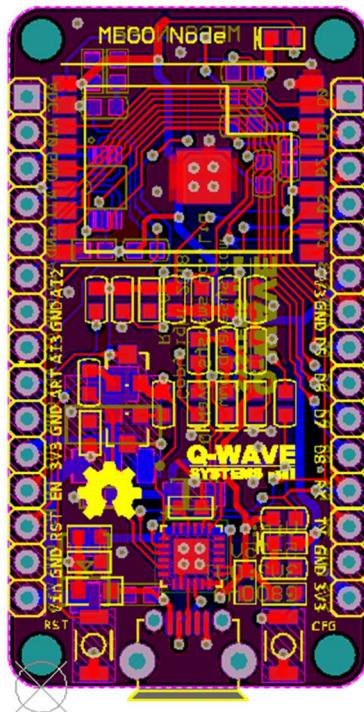
บอร์ด MEGO Node มี 2 เวอร์ชันคือ Model A และ B



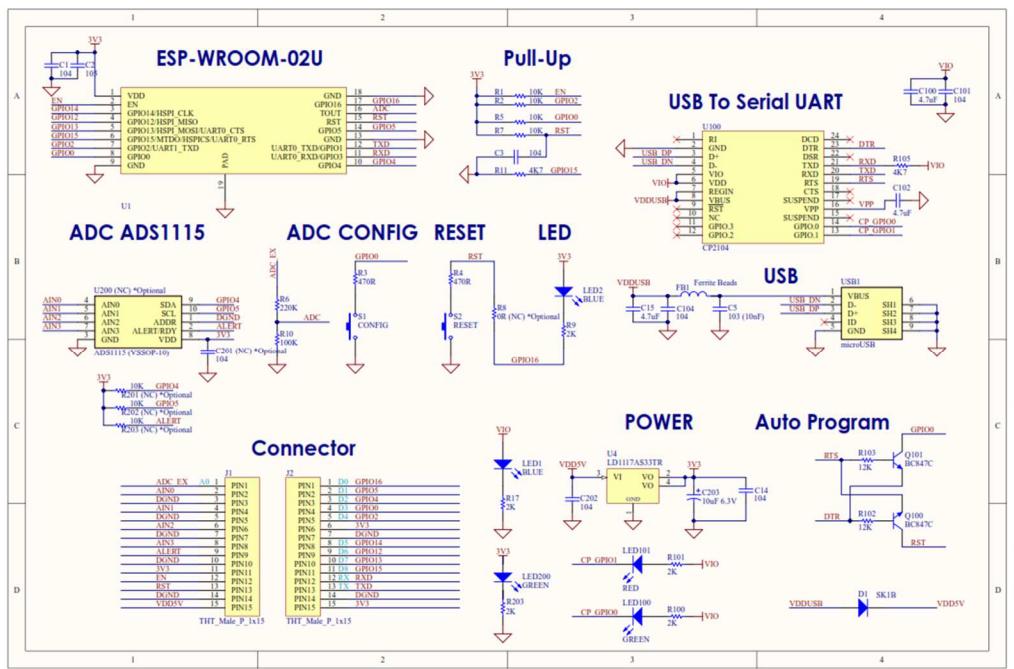
การจัดวาง Component และ PCB Layout

สามารถดาวน์โหลดไฟล์ออกแบบได้ที่นี่

[https://github.com/QWaveSystems/MEGO\\_Node](https://github.com/QWaveSystems/MEGO_Node)



การอธิบายบังคับ MEGO Node (Schematic)



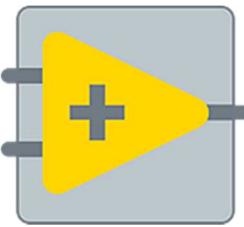
# Connector

ADC	EX	A0	1	J1		J2	
AIN0			2	PIN1		PIN1	1 D0 GPIO16
DGND			3	PIN2		PIN2	2 D1 GPIO5
AIN1			4	PIN3		PIN3	3 D2 GPIO4
DGND			5	PIN4		PIN4	4 D3 GPIO0
AIN2			6	PIN5		PIN5	5 D4 GPIO2
DGND			7	PIN6		PIN6	6 3V3
AIN3			8	PIN7		PIN7	7 DGND
ALERT			9	PIN8		PIN8	8 D5 GPIO14
DGND			10	PIN9		PIN9	9 D6 GPIO12
3V3			11	PIN10		PIN10	10 D7 GPIO13
EN			12	PIN11		PIN11	11 D8 GPIO15
RST			13	PIN12		PIN12	12 RXD
DGND			14	PIN13		PIN13	13 TX TXD
VDD5V			15	PIN14		PIN14	14 DGND
				PIN15		PIN15	15 3V3

สามารถดาวน์โหลดไฟล์เอกสารแบบได้ที่นี่

[https://github.com/QWaveSystems/MEGO\\_Node](https://github.com/QWaveSystems/MEGO_Node)

## การติดตั้งโปรแกรม LabVIEW

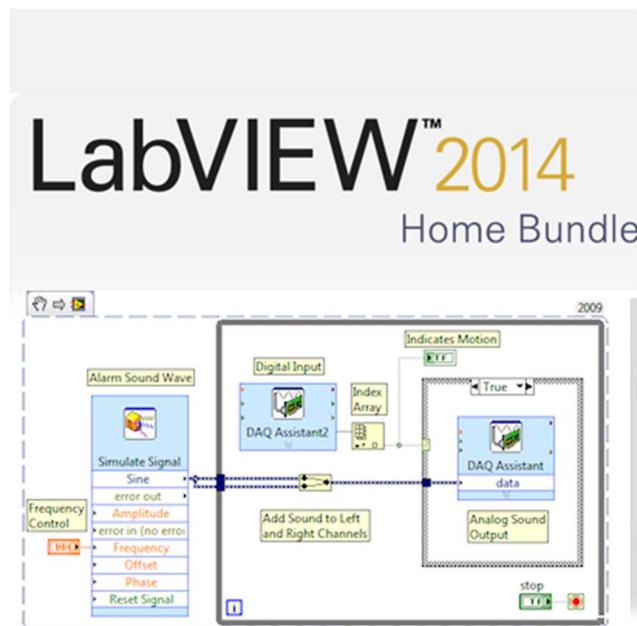


LabVIEW เป็นภาษาโปรแกรมในรูปแบบกราฟฟิก โดยนักสามารถออกแบบพัฒนาโปรแกรมด้วยการลากวางบล็อกฟังก์ชันต่าง ๆ ประกอบกันเป็นแพล็อกซ์ชัน โดยบอร์ด MEGO รองรับ LabVIEW เวอร์ชัน 2014 ขึ้นไป

### ติดตั้ง LabVIEW 2014 Home Edition

บริษัท National Instruments เสิร์ฟเวอร์ตลาด นักเรียน นักศึกษา Home Users และ Maker Users ก็มีความต้องการใช้งาน LabVIEW แต่ติดกับราคายกของ License แบบ Commercial สูงเกินเอื้อม จึงได้ออก พลิตกับนักที่ใหม่ (Software License) ที่มีชื่อว่า "LabVIEW Home Edition" หรือชื่อในการทำตลาดคือ "LabVIEW Home Bundle for Windows" โดยจะมีเวอร์ชันที่รันได้บน Windows เก่าแก่ โดยเวอร์ชัน สูงสุดก็คือ LabVIEW 2014 Service Pack 1 (32bit)

โดยเจื่อนในการใช้งาน LabVIEW Home Edition นั้นจะยังยอมให้เพื่อการเรียนรู้ (Home) หรือ พัฒนาส่วนบุคคลเก่าแก่ (Maker) ห้ามมิให้ใช้งานในเชิงพาณิชย์ ในสถาบันศึกษา หรือในงานอุตสาหกรรม คือ เป็น License เพื่อใช้งานส่วนตัว ตามชื่อ Home Edition



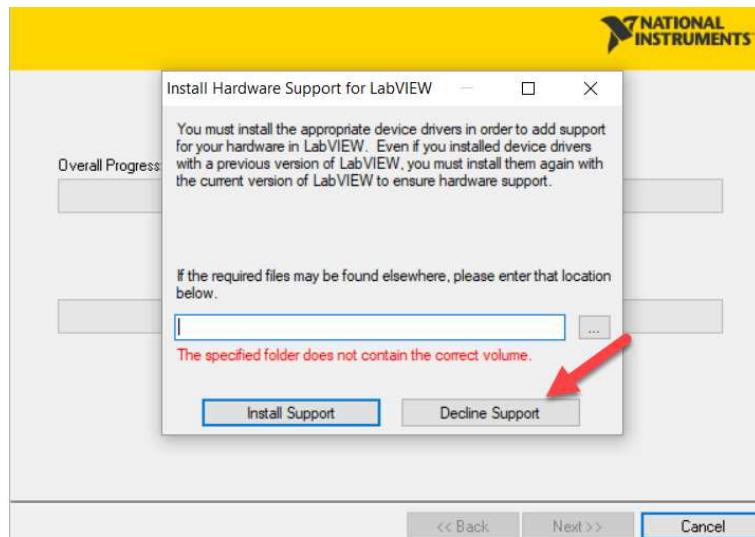
LabVIEW Home Edition ประกอบไปด้วยฟังก์ชันต่าง ๆ ดังต่อไปนี้

- LabVIEW Full Development System
- Application Builder (Build RT App, EXE, Installer)
- LabVIEW Control Design and Simulation Module
- LabVIEW MathScript RT Module

สำหรับผู้ที่ต้องการใช้งานเชิงการค้า (Commercial) จะต้องใช้เวอร์ชันล่าสุด โดย License ที่รองรับบอร์ด MEGO ได้คือ Base Edition ขึ้นไป

ลิงค์ดาวน์โหลด LabVIEW 2014 (32bit) โดยสามารถทดลองใช้งาน ได้ 7-45 วัน  
<http://download.ni.com/evaluation/labview/ekit/other/downloader/2014sp1LV-WinEng.exe>

โดยขั้นตอนการติดตั้งจะใช้เวลา 15-30 นาที ขึ้นอยู่กับความเร็วของคอมพิวเตอร์ เมื่อติดตั้งเสร็จ สิ้น โปรแกรมจะถามว่าต้องการติดตั้ง Driver Support เพิ่มหรือไม่ ให้กดเลือก "Decline Support" ตามรูป จากนั้น ให้ Re-Start เครื่องคอม 1 รอบ เป็นอันเสร็จสิ้นขั้นตอนการติดตั้ง LabVIEW ครับ



ในการนี้ที่เป็นนักเรียน นักศึกษา สามารถลงทะเบียนเพื่อใช้งานฟรี ระยะเวลา 6 เดือน โดยกรอกข้อมูลเกี่ยวกับสถานศึกษาที่กำลังศึกษาอยู่ โดยคลิกลงทะเบียน Free-6-Month-Evaluation-of-LabVIEW-Student-Edition-for-at-home ได้ที่ลิงค์นี้

<https://forums.ni.com/t5/Archive-TKB/Free-6-Month-Evaluation-of-LabVIEW-Student-Edition-for-at-home/ta-p/3497362>

ติดตั้ง Driver NI-VISA (Universal I/O Interface Software)

โดย NI-VISA คือ The Virtual Instrument Software Architecture (VISA) เป็น Driver กี่จะทำให้เราสามารถดาวน์โหลดโปรแกรมลงบอร์ด MEGO ผ่านทาง COM Port (RS-232) หรือ รับ-ส่งข้อมูลผ่านทาง COM Port ระหว่างบอร์ด MEGO และ คอมพิวเตอร์ได้

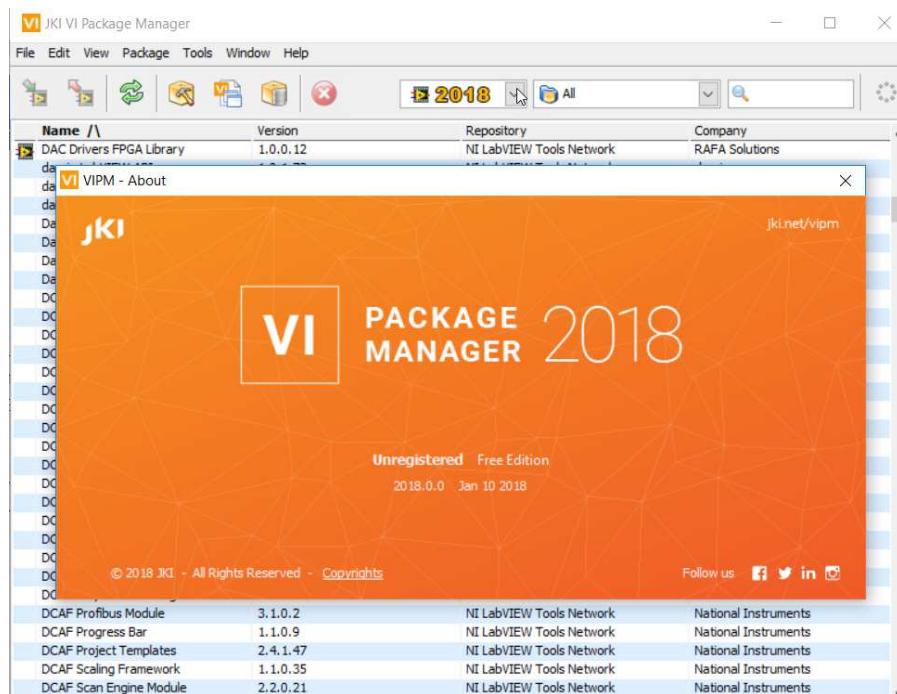
สำหรับผู้ใช้ LabVIEW 2014 ให้ติดตั้ง NI-VISA 14.0.1 (\*สำหรับผู้ที่ใช้งาน LabVIEW เวอร์ชันล่าสุดให้ติดตั้ง NI VISA ตามเวอร์ชันของ LabVIEW ที่ใช้งาน)

<http://download.ni.com/support/softlib/visa/NI-VISA/14.0.1/NIVISA1401full.exe>

ติดตั้งโปรแกรม “VI Package Manager”

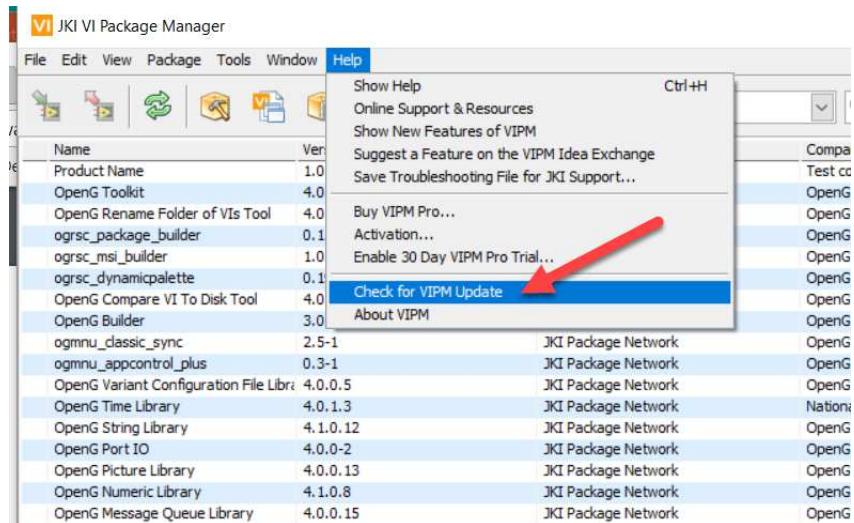
โปรแกรม “VI Package Manager” เป็นโปรแกรมเสริมที่ทำหน้าที่จัดการติดตั้ง Addons, 3rd Party Toolkit ภายใน LabVIEW ก็ง่ายดาย (พัฒนาโดย jki.net) โดยเราจะใช้โปรแกรม VIPM นี้ติดตั้ง MEGO Driver ให้ใช้งานร่วมกับ LabVIEW โดยเวอร์ชันที่ใช้ร่วมกับ MEGO Driver (.vip) ได้บันทึกต้องเป็นเวอร์ชัน 2018 หรือสูงกว่าเท่านั้น

สำหรับผู้ที่ใช้งาน LabVIEW เวอร์ชัน 2018 หรือสูงกว่า ไม่จำเป็นต้องติดตั้งเพิ่มเติม เพราะว่า VI Package Manager ติดตั้งมาพร้อมกับ LabVIEW 2018 สามารถเปิดโปรแกรม VI Package ได้เรียบร้อยแล้ว

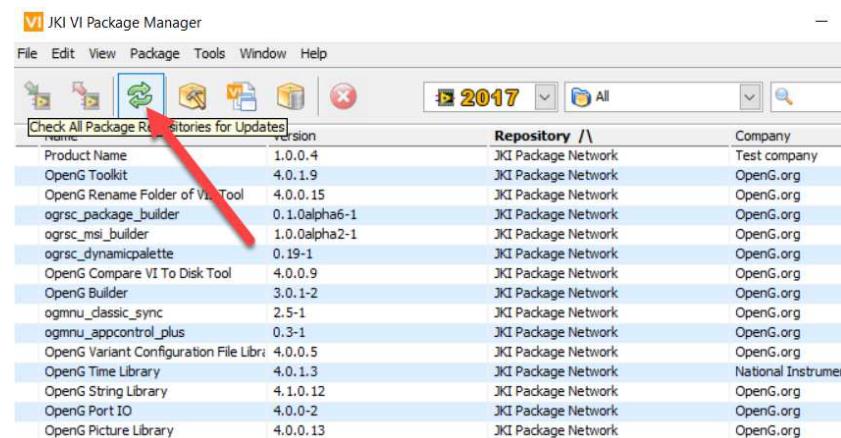


สำหรับผู้ที่ใช้งาน “LabVIEW 2014” ต้องทำการอัพเกรด จาก “VI Package Manager 2014” เป็น 2018 ตามขั้นตอนนี้

วิธีการ Update อัตโนมัติ เมื่อเปิดโปรแกรม VI Package Manager ขึ้นมาแล้วให้กด “Help > Check for VIPM Update” จากนั้นโปรแกรมจะดาวน์โหลดอัตโนมัติ และติดตั้งเวอร์ชันล่าสุดให้อัตโนมัติ



เมื่อเริ่มต้นใช้งานครั้งแรก เชื่อมต่อคอมพิวเตอร์กับอินเทอร์เน็ต จากนั้นให้กด “Check All Package for Update” เพื่อเชื่อมต่อ กับ Server เพื่ออัปเดตโปรแกรมต่าง ๆ ขั้นตอนนี้จะใช้เวลาประมาณ 5-10 นาที ขึ้นอยู่กับความเร็วอินเทอร์เน็ต

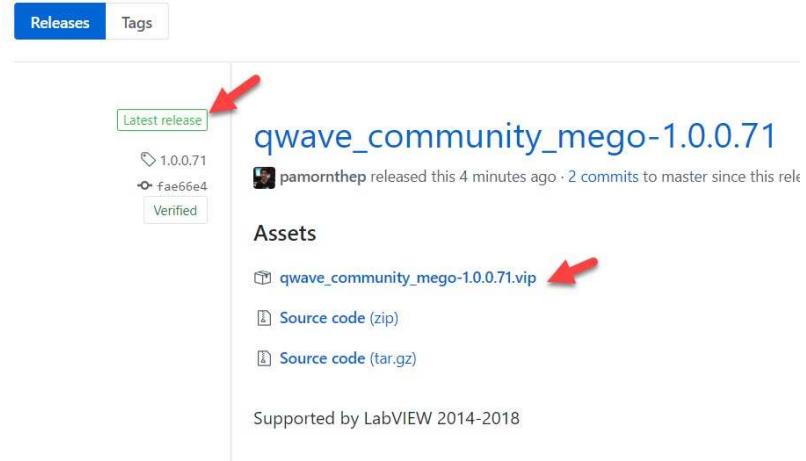


ติดตั้ง MEGO Driver (\*.vip)

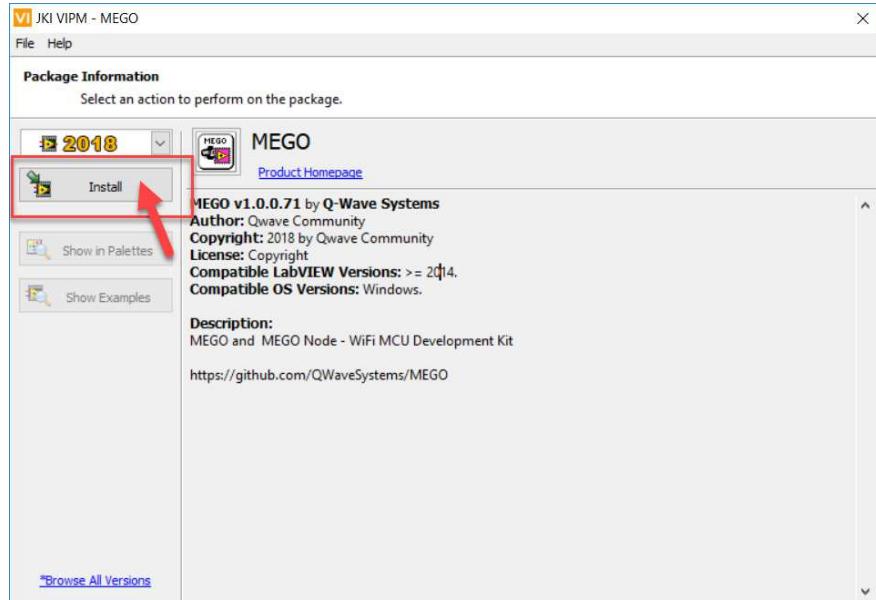
โดยไฟล์ Driver จะมีชื่อว่า “qwave\_community\_mego-1.0.0.xx.vip” สามารถดาวน์โหลดได้ที่นี่



[https://github.com/QWaveSystems/MEGO\\_MEGO-Node\\_LabVIEW-Driver/releases](https://github.com/QWaveSystems/MEGO_MEGO-Node_LabVIEW-Driver/releases)

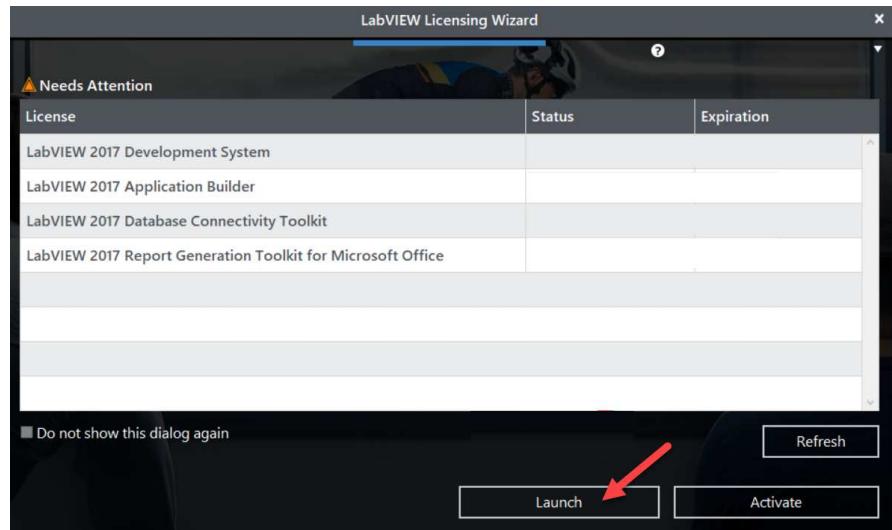


ดับเบิลคลิก เปิดไฟล์ MEGO Driver ที่มีนามสกุล “.vip” จากนั้นโปรแกรม “VI Package Manager” จะเปิดขึ้นมาอัตโนมัติ เพื่อเริ่มขั้นตอนการติดตั้งและดังรูป

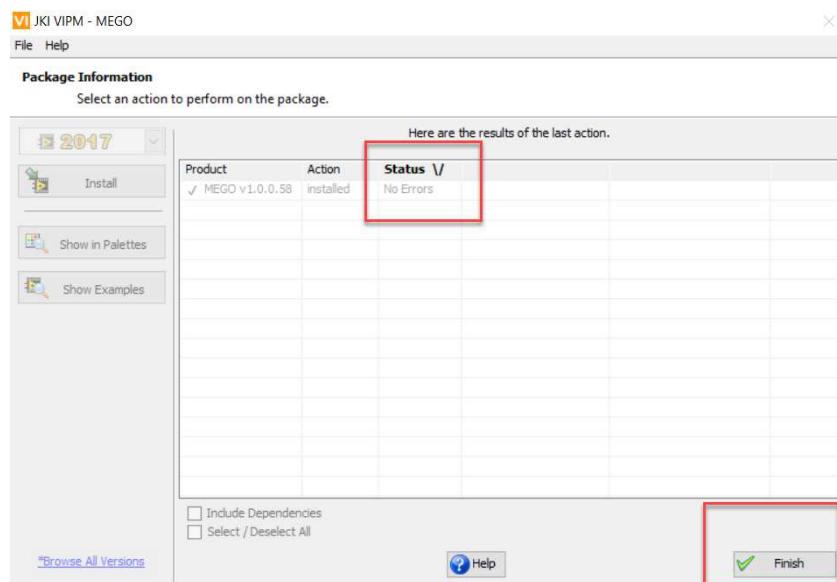


เมื่อกด Install โปรแกรม VI Package Manager จะเปิดโปรแกรม LabVIEW ขึ้นมาอัตโนมัติ เพื่อเริ่มการติดตั้ง MEGO Driver ลงใน LabVIEW เวอร์ชันที่กำหนด

ในการนี้กีบ่างเครื่องที่ติดตั้งครั้งแรก ในขั้นตอนนี้จะโปรแกรม LabVIEW และจะค้างอยู่กีต่อang "License Wizard" หน้าแรกดังนี้ ให้กด "Launch" เพื่อเปิดโปรแกรม จากนั้น MEGO Driver จะติดตั้งอัตโนมัติ



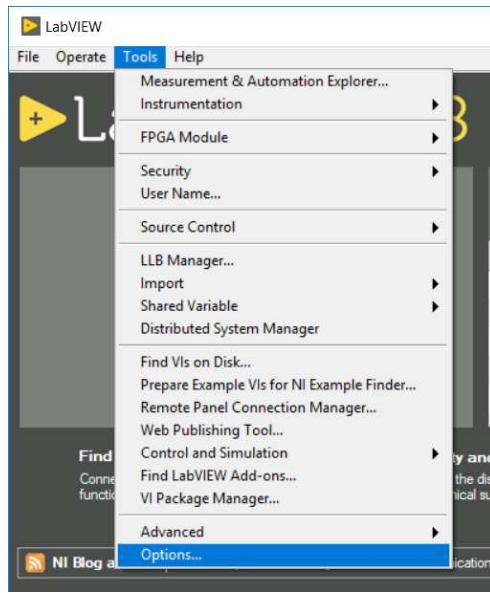
ในขั้นตอนสุดท้ายของการติดตั้งให้ตรวจสอบ Status แสดงดังรูป ในกรณีที่ติดตั้งสมบูรณ์จะต้องขึ้นว่า "No Errors"



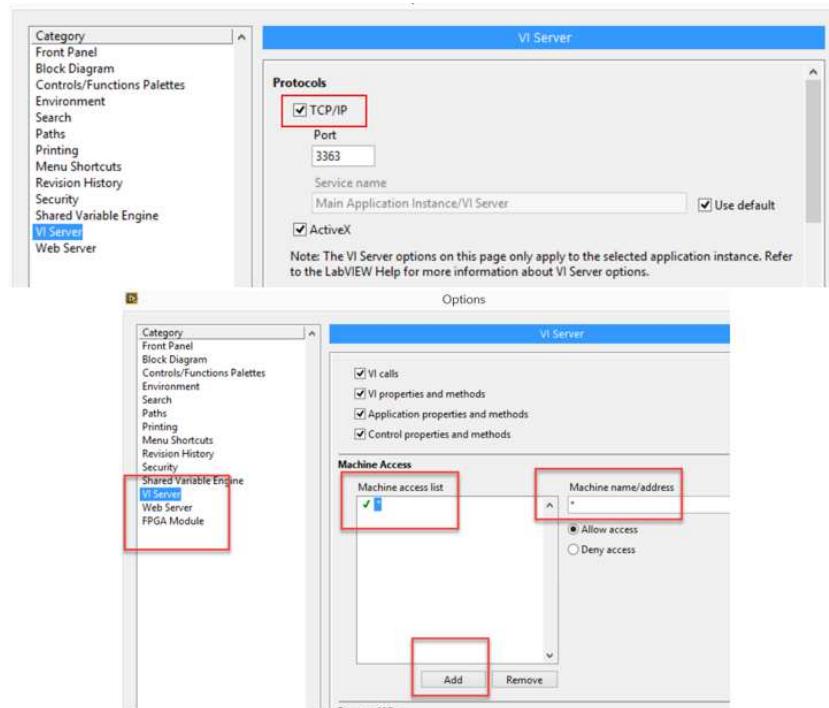
ในกรณีที่มี Error เกิดขึ้นในขั้นตอนสุดท้าย หรือติดตั้งนานกว่าปกติ จนเกิด Timeout แนวทางแก้ไขปัญหาดังนี้

ติดตั้งไม่ได้ หรือไม่มีปุ่ม Install ขึ้นมา ให้ตรวจสอบ Path ที่เก็บไฟล์ MEGO Driver (\*.vip) ถ้ามีข้อความภาษาไทย (\* เช่น C:\Users\Admin\Desktop\ทดสอบ\\*.vip) จะทำให้การติดตั้งมีปัญหา ให้เปลี่ยน Path เป็นภาษาอังกฤษเท่านั้น

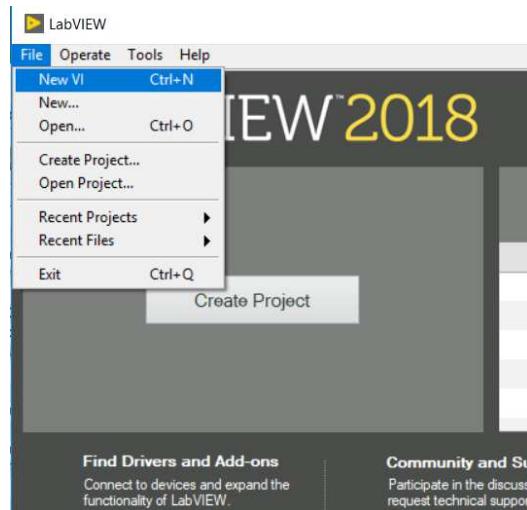
ติดตั้งนานกว่าปกติ จนเกิด Timeout สาเหตุนี้เกิดจาก โปรแกรม VI Package Manager ไม่สามารถเชื่อมต่อ กับ LabVIEW ได้เนื่องจาก TCP Setting ของเครื่องนั้น ๆ ไม่ยอมให้โปรแกรมสื่อสารกัน ผ่านพอร์ตที่กำหนด แนวทางแก้ไขคือ ให้เปิดโปรแกรม LabVIEW จากนั้นเลือก (Tool > Options)



เลือก "VI Server" เพิ่มเครื่องหมาย "\*" ในช่อง "Machine Name/Address" กด Add จากนั้น กลับไปติดตั้ง MEGO Driver 亟ครั้ง

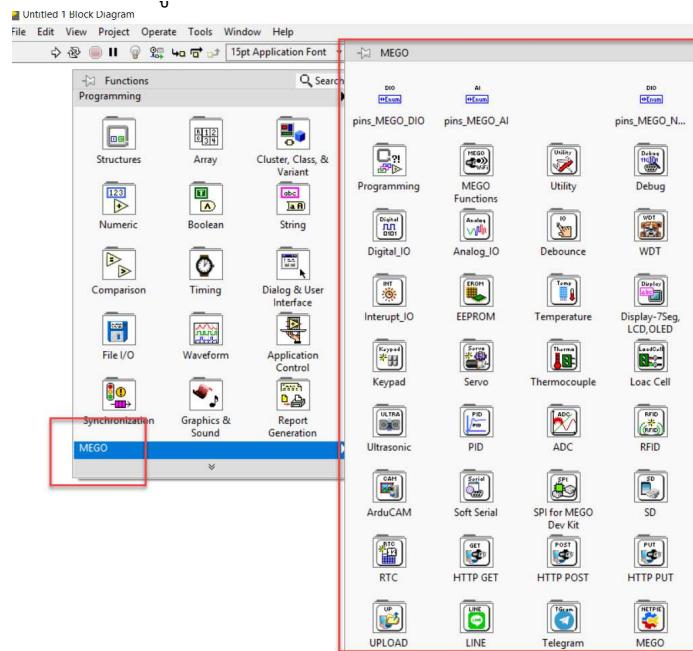


ตรวจสอบ MEGO Driver ที่ติดตั้งเรียบร้อยแล้ว



เปิดโปรแกรม LabVIEW จากนั้นเลือก "File > New VI" โดยโปรแกรมจะเปิด 2 หน้าต่างขึ้นมา หน้าต่าง "Front Panel" (สีเทา) และหน้าต่าง Block Diagram (สีขาว) ในการเลือกระหว่าง 2 หน้าต่างนี้ ให้กด Shortcut "Ctrl+E"

โดยในหน้าต่าง Block Diagram (พื้นที่สีขาว) ให้คลิกขวา จะเห็น Function Palette "MEGO" แสดงดังรูป แสดงว่าการติดตั้งสมบูรณ์



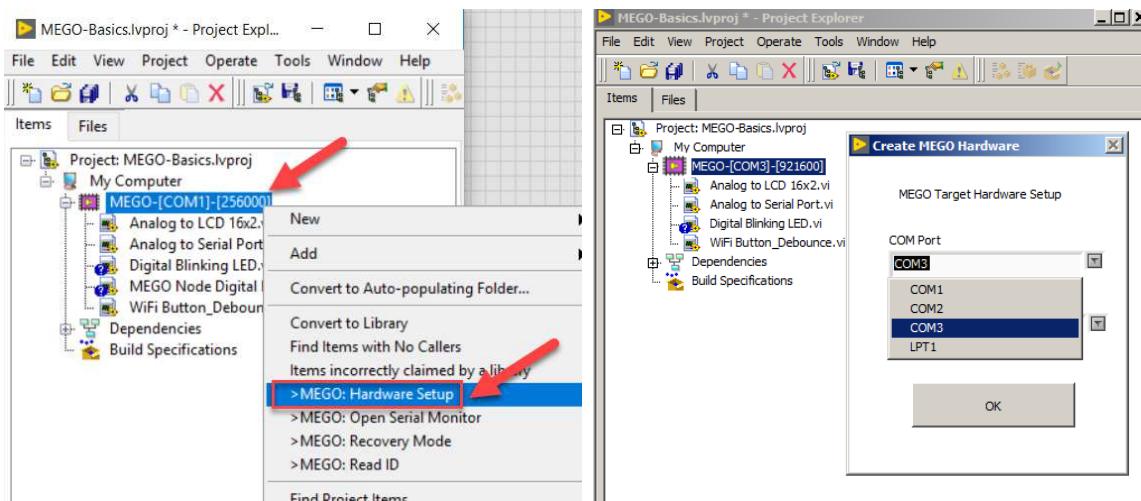
จากนั้น ตรวจสอบไฟล์ตั้งอย่างทั้งหมดได้ที่นี่ "C:\Program Files (x86)\National Instruments\LabVIEW 2018\vi.lib\Q-Wave Systems\MEGO\Examples" ถ้ามีไฟล์ดังรูปแสดงว่าการติดตั้งสมบูรณ์

Name	Date modified	Type	Size
_MEGO Design Template	7/14/2018 4:24 PM	File folder	
7Segment	7/14/2018 4:24 PM	File folder	
ADC	7/14/2018 4:24 PM	File folder	
Basics	7/14/2018 4:24 PM	File folder	
Data Logger	7/14/2018 4:24 PM	File folder	
ESP8266 Utility	7/14/2018 4:24 PM	File folder	
Hardware Interface	7/14/2018 4:24 PM	File folder	
Keypad	7/14/2018 4:24 PM	File folder	
LINE Notify	7/14/2018 4:24 PM	File folder	
MicroGear	7/14/2018 4:24 PM	File folder	
NETPIE Feed	7/14/2018 4:24 PM	File folder	
OLED	7/14/2018 4:24 PM	File folder	
PID	7/14/2018 4:24 PM	File folder	
Sensors	7/14/2018 4:24 PM	File folder	
SHT1X	7/14/2018 4:24 PM	File folder	

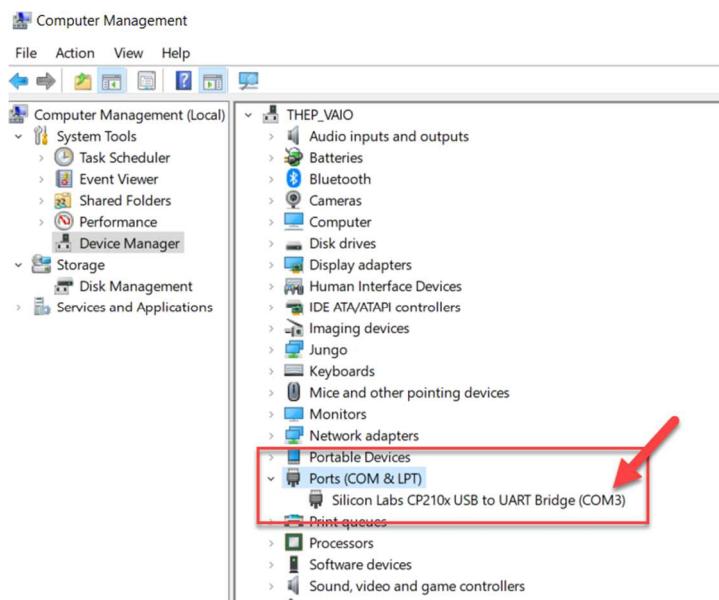
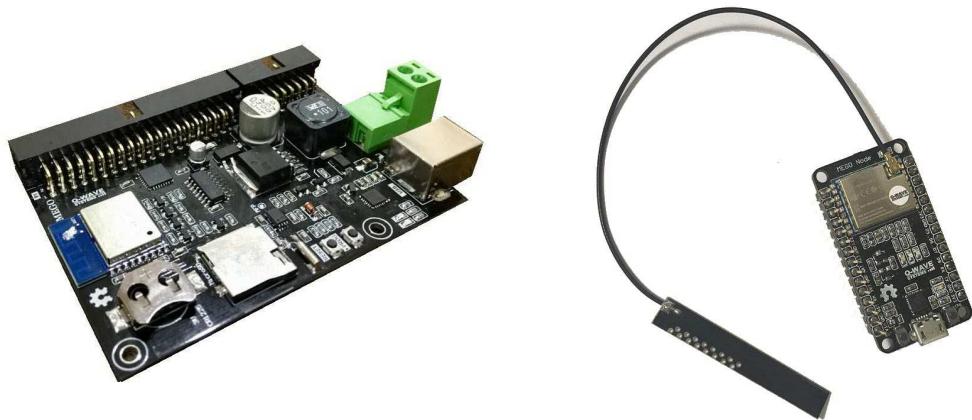
กดสอบสร้างโปรแกรม LabVIEW เบื้องต้น

เปิดตัวอย่างโปรแกรม โดยเข้าไปที่ Folder “C:\Program Files (x86)\National Instruments\LabVIEW 2018\vi.lib\Q-Wave Systems\MEGO\Examples\Basics” จากนั้นเปิดไฟล์ “MEGO-Basics.lvproj”

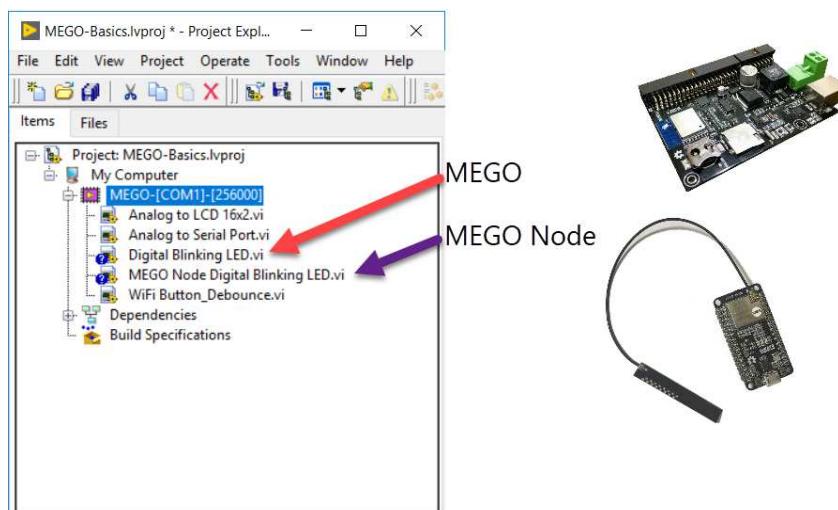
จากเบื้องต้นคลิกขวาที่ “MEGO-[COMx]-[25600]” เลือก “>MEGO : Hardware Setup”



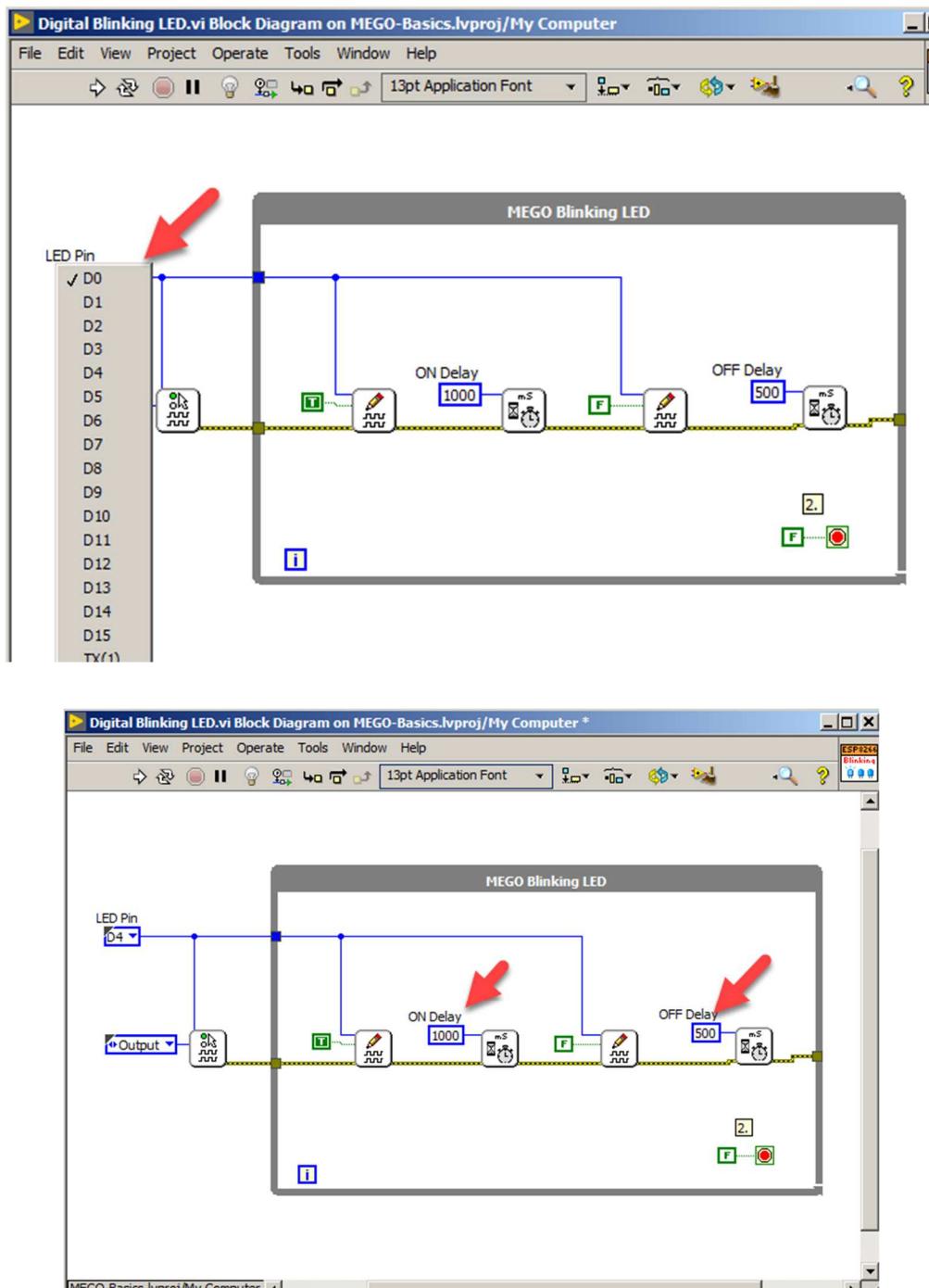
ในขั้นตอนนี้ต้องเสียบสาย USB ของบอร์ด MEGO เข้ากับคอมพิวเตอร์ จากนั้นตรวจสอบว่า MEGO ต่ออยู่ที่ COM Port ที่เท่ากับ 921600 Buad ให้ถูกต้อง เลือก Upload Speed ที่ต้องการ โดยบอร์ดรองรับ Speed สูงสุดคือ 921600 Buad



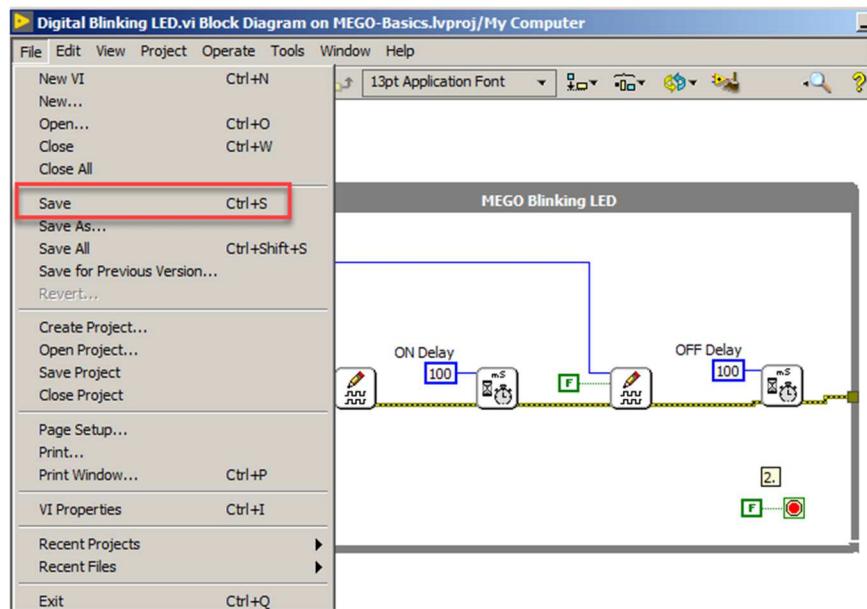
จากนั้นในหน้าต่าง Project Explorer ในกรณีที่ใช้บอร์ด MEGO ให้ดับเบิลคลิกก็ไฟล์ “Digital Blinking LED.vi” ในกรณีที่ใช้บอร์ด MEGO Node ให้ดับเบิลคลิกก็ไฟล์ “MEGO Node Digital Blinking LED.vi” ซึ่งเป็นโปรแกรมไฟกระพรุน (Blinking LED) ที่ Pin LED บนบอร์ด



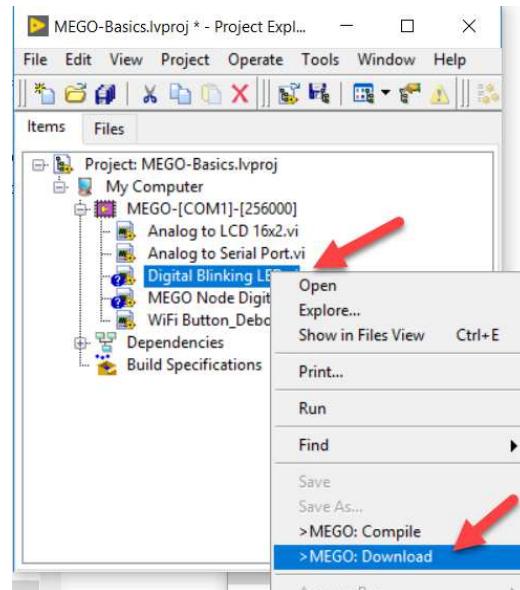
เมื่อเปิดโปรแกรมขึ้นมา จะเห็น Code ของโปรแกรม Blinking LED จากนั้นคลิกที่ LED Pin ให้เลือก Pin ที่ต่อไว้กับ LED ที่อยู่บนบอร์ด MEGO สำหรับบอร์ด MEGO ไม่มี LED ดังนั้นต้องต่อสายไฟ และ LED บนบอร์ด I/O ด้วยตนเอง แต่บอร์ด MEGO Node สามารถกำหนด Pin 16 ได้เลย



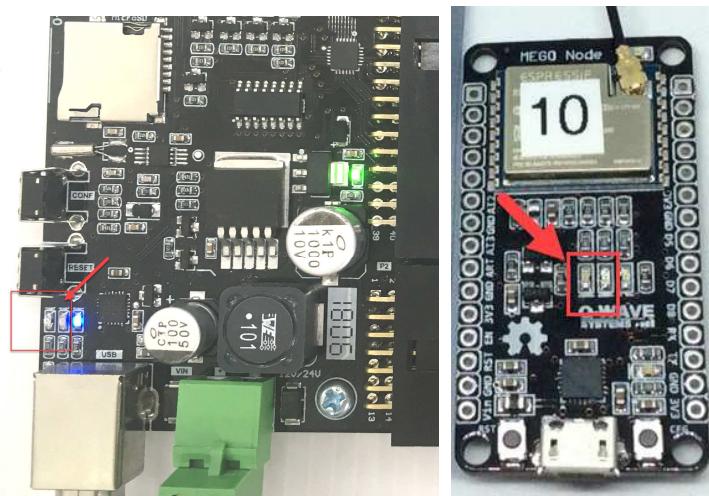
โดยสังเกตว่าตัวแปร "ON Delay" และ "OFF Delay" เป็นตัวกำหนดระยะเวลาที่ทำให้ LED ติดดับ สามารถทดลองเปลี่ยนค่าได้ตามต้องการ จากนั้นให้กด Save โปรแกรม



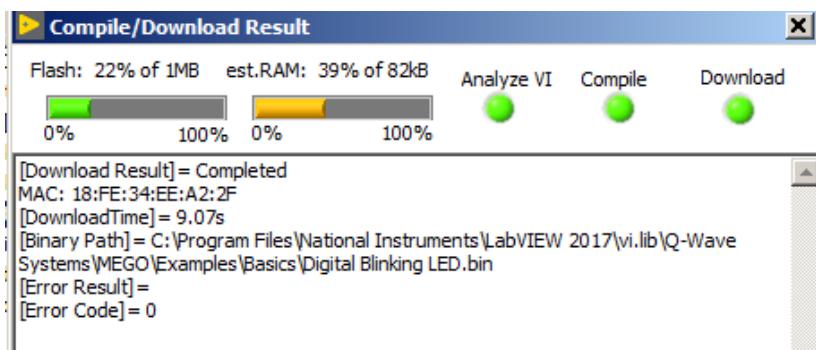
จากนั้นให้กดลง “Download” ลงบอร์ด MEGO โดยการคลิกข้ามที่โปรแกรม “Digital Blinking LED.vi” เลือก Download



ชั้งขั้นตอนการ Download จะมีการ Compile ใหม่ทุกครั้ง ก้าวไฟส์ vi มีการแก้ไข (ในกรณีไฟส์ vi ไม่มีการแก้ไขโปรแกรมก็จะ Download กันก็) จากนั้นรอสักครู่สังเกตที่บอร์ด MEGO จะมี LED สีฟ้า (TX) และสีแดง (RX) ที่แสดงผลการโปรแกรมบนบอร์ด โดยจะรับประมวลผลการโปรแกรม



เมื่อโปรแกรมเสร็จสมบูรณ์จะแสดงผลการ Compile หรือ Download แสดงดังนี้



ผลของการ Compile หรือ Download มีความหมายดังนี้

Flash % จะแสดงขนาดพื้นที่ของ Flash Memory ที่เก็บโปรแกรมบันช์ฟายในครอคุนໂກຣລອ່ວນ MCU โดยจากการ Compile นี้แสดงให้เห็นว่ากินพื้นที่ขนาด 22% จากทั้งหมด 1MB

Estimate RAM % จะแสดงพื้นที่ของหน่วยความจำ RAM ที่คาดว่าจะใช้งานของโปรแกรม โดยค่านี้เป็นค่าที่ประมาณการ โดยคิดคำนวนจากตัวแปรต่าง ๆ ที่สร้างขึ้นในโปรแกรม โดยผลจากการ Compile นี้แสดงให้เห็น กินพื้นที่หน่วยความจำประมาณ 39% ของ 82kB ของ RAM ทั้งหมด

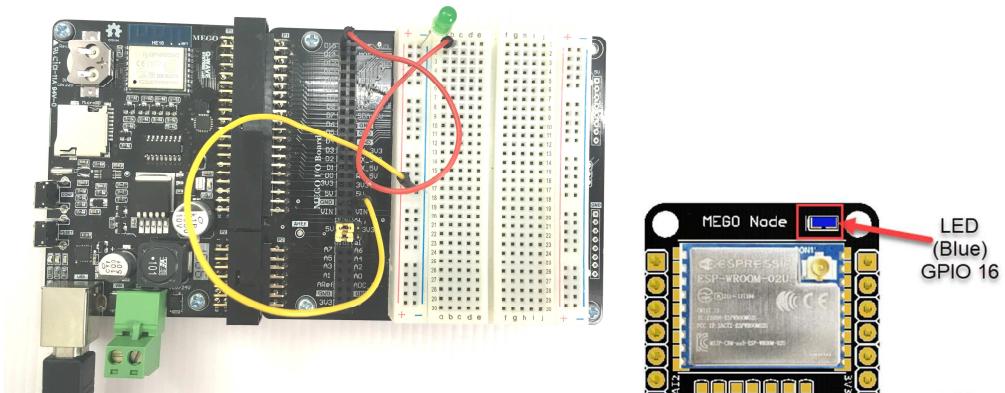
MAC ID เป็นหมายเลขเฉพาะของบอร์ด MEGO โดยว้างอิงจาก MAC Address ของซีพีไอในครอคุนໂກຣລອ່ວນ MCU อาทิ 18:FE:34:EE:A2:2F

Binary File Path เป็นโปรแกรมที่ Compile เรียบร้อยแล้ว สามารถตรวจสอบไฟล์ “Digital Blinking LED.bin” ซึ่งเป็น Binary ไฟล์ที่เกิดจากการ Compile โดยไฟล์นี้จะเป็นโปรแกรมที่สร้างขึ้นในขั้นตอน Compile จากนั้นในขั้นตอนการ Download จะทำการนำไฟล์นี้ไปโปรแกรมลงบอร์ด MEGO

Checksum เป็นไฟล์ที่เกิดขึ้นจากขั้นตอน Compile โดยเป็น Checksum ของ โปรแกรม .vi ณ เวลาที่ Compile ล่าสุด โดยเอาไว้ตรวจสอบในขั้นตอนดาวน์โหลด โดยถ้าไฟล์ Checksum ไม่เปลี่ยนแปลงก็ ให้ทำการ Download Bin ไฟล์เดิมลงบอร์ดได้กันที แต่ถ้าไฟล์ Checksum มีการเปลี่ยนแปลงแสดงว่าต้อง Compile โปรแกรมใหม่ จากนั้นค่อย Download

Name	Date modified	Type
Analog to LCD 16x2.vi	5/29/2018 5:45 PM	LabVIEW I
Analog to Serial Port.vi	5/29/2018 5:45 PM	LabVIEW I
Digital Blinking LED.bin	5/29/2018 10:49 PM	BIN File
Digital Blinking LED.checksum	5/29/2018 10:49 PM	CHECKSUM
Digital Blinking LED.vi	5/29/2018 5:45 PM	LabVIEW I
MEGO-Basics.aliases	5/30/2018 12:22 AM	ALIASES Fi
MEGO-Basics.lvlp	5/30/2018 12:22 AM	LVLPS File
MEGO-Basics.lvproj	5/30/2018 12:22 AM	LabVIEW F
WiFi Button_Debounce.vi	5/29/2018 5:45 PM	LabVIEW I

ในขั้นตอนนี้ให้สังเกตผลของ LED ที่ต่อ กับบอร์ด MEGO แสดงว่าการติดตั้ง Software และ MEGO Driver สมบูรณ์พร้อมใช้งาน



รูปชี้ราย บอร์ด MEGO ต่อ LED ที่ D15 และ GND  
รูปขวา บอร์ด MEGO Node LED ที่ D0 (Pin16)

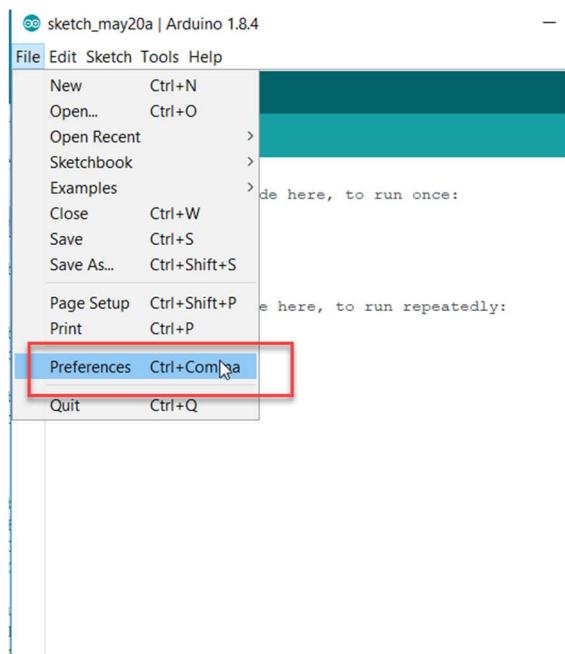
## การใช้งาน MEGO ด้วยภาษา C/C++

สำหรับผู้ที่ต้องการพัฒนาโปรแกรมบนบอร์ด MEGO ด้วยภาษา C/C++ สามารถทำการติดตั้ง Arduino Board Package ตามขั้นตอนต่อไปนี้

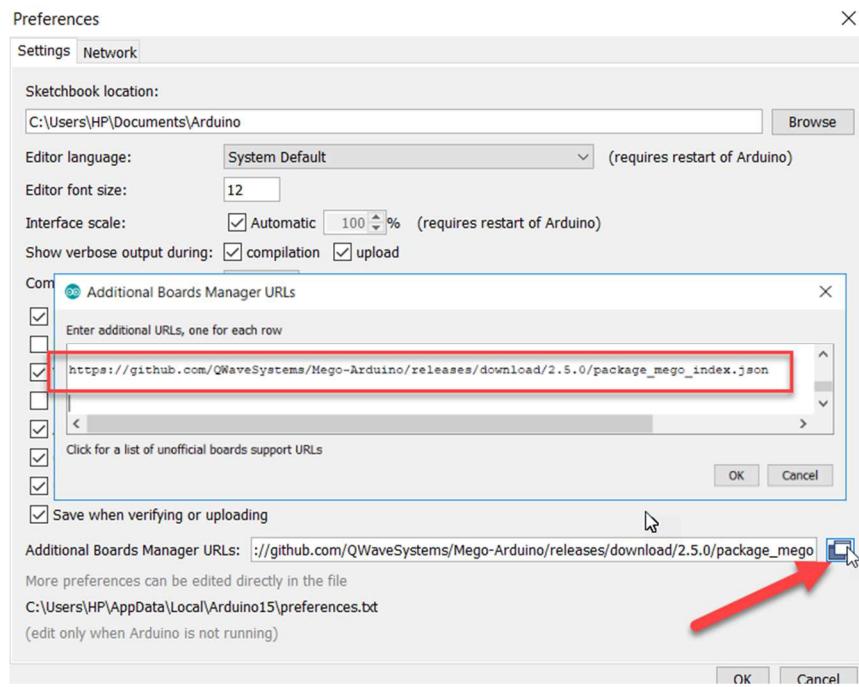
### Download the Arduino IDE



ติดตั้ง Arduino IDE (\*เวอร์ชันล่าสุด ณ วันที่เขียนบทความนี้คือ เวอร์ชัน 1.8.5) ดาวน์โหลดได้จากที่นี่ <https://www.arduino.cc/en/Main/Software>

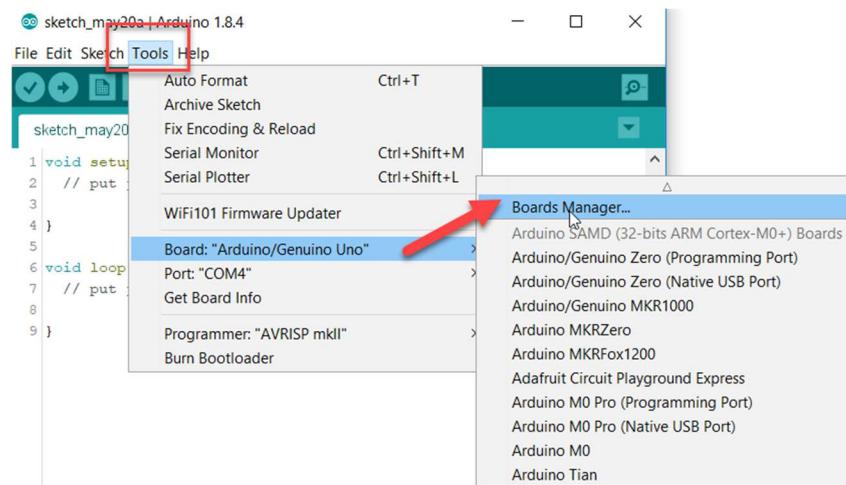


เปิดโปรแกรม Arduino IDE ขึ้นมา จากนั้นเลือก File > Preferences เพื่อตั้งค่า

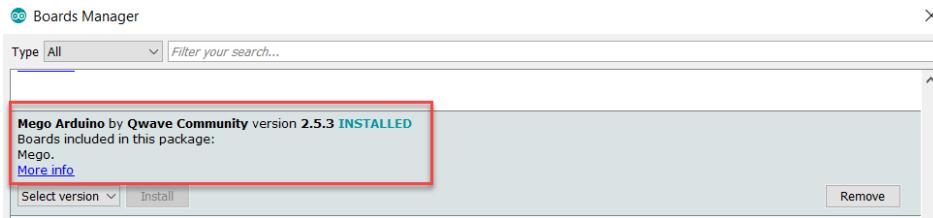


ໃນຫຼັບຂ້ອງ “Additional Boards Manager URLs” ຄລືກເພື່ອເພີ່ມ Board Package ໂດຍເພີ່ມລົງຄໍ MEGO Board ໂດຍພິມໄວ້ .json ດັ່ງນີ້

[https://github.com/QWaveSystems/Mego-Arduino/releases/download/package\\_mego\\_index.json](https://github.com/QWaveSystems/Mego-Arduino/releases/download/package_mego_index.json)

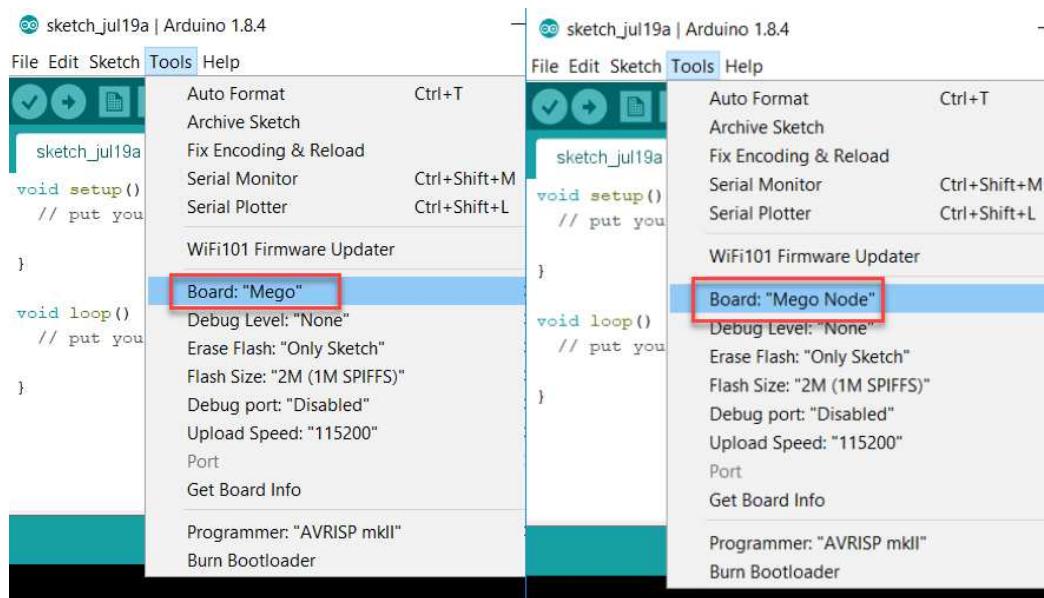


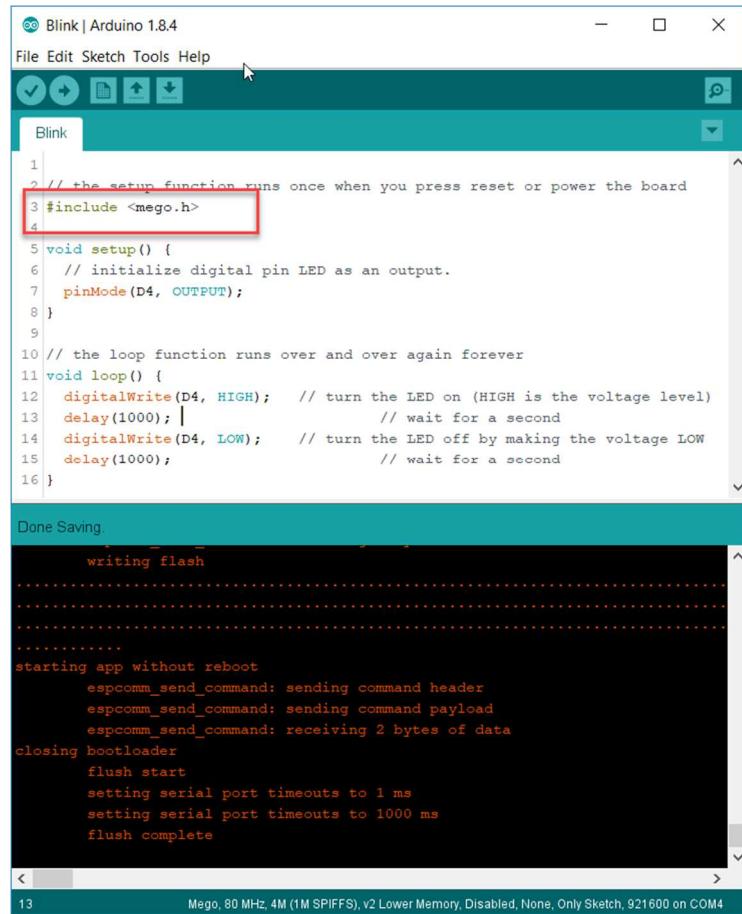
ຈາກບັນ ໃກລົກກໍ “Tools > Board > Board Manager” ເພື່ອຕົດຕັ້ງ MEGO Board Package ໂດຍຈະໃຫ້ວລາສັກຄຽນໃນການວັບແດຈ ໃນຂັ້ນຕອນນັ້ນຕ້ອງເຊື່ອມຕ່ອງອົນເກອຮ໌ເນື້ຕ



ຈະສັງເກຕາເໜີວ່າ ຈະມີ Package ດັ່ງນີ້ໃຫ້ເລືອກ "Mego Arduino by Qwave Community" ຈາກບັນຄຶກ Install ການຕັດຕັ້ງຈະໃໝ່ວລາສັກຄູ ເນື່ອຈາກວ່າຕ້ອງດາວໂຫລດໄວ້ ປະມານ 150 MB ເຊິ່ງ ຕັດຕັ້ງເສັ້ນສົມບຸຮນ໌

### ການໃໝ່ງານໃຫ້ເລືອກບ່ອນດໍາ MEGO ກັບ MEGO Node ແລ້ວດັ່ງນີ້





การเขียนโปรแกรมร่วมกับบอร์ด MEGO ต้องเพิ่ม “`#include <mego.h>`” ทุกครั้งเมื่อจะเขียน Sub-Function ที่กำหนดการ GPIO ॥ 亦 ADC บนบอร์ด MEGO

```

#include <mego.h>

void setup() {

    pinMode(D4, OUTPUT);
}

void loop() {
    digitalWrite(D4, HIGH);
    delay(1000);
    digitalWrite(D4, LOW);
    delay(1000);
}

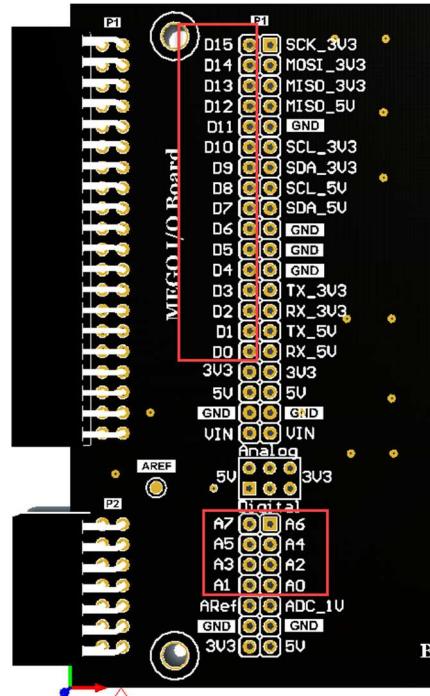
```

ตัวอย่างโปรแกรม ไฟกระพริบที่ Pin = "D4"

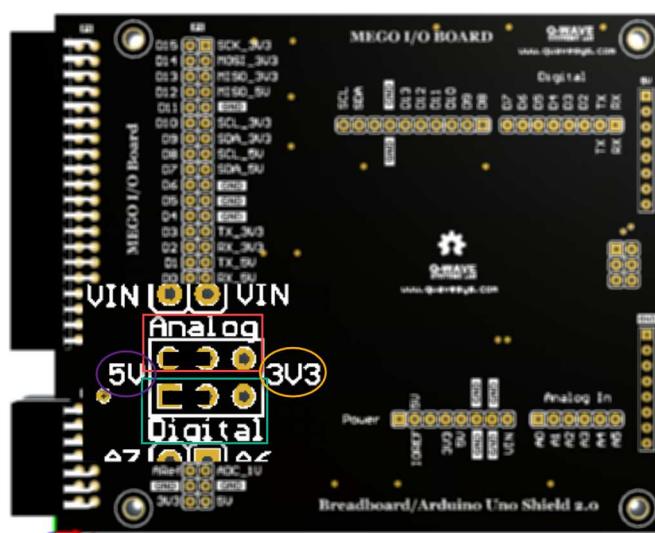
สำหรับบอร์ด MEGO การกำหนด Digital/Analog Pin เพื่อเรียกใช้งาน สามารถกำหนดได้โดยตรงดังนี้

Digital Pins : D0-D15 (16 Channels)

Analog Pins : A0-A7 (8 Channels)



โดยทั้ง Digital/Analog Pin สามารถกำหนด Level ที่ใช้งานได้ทั้ง 3.3 V และ 5 V โดยกำหนดที่ Jumper แสดงดังรูป



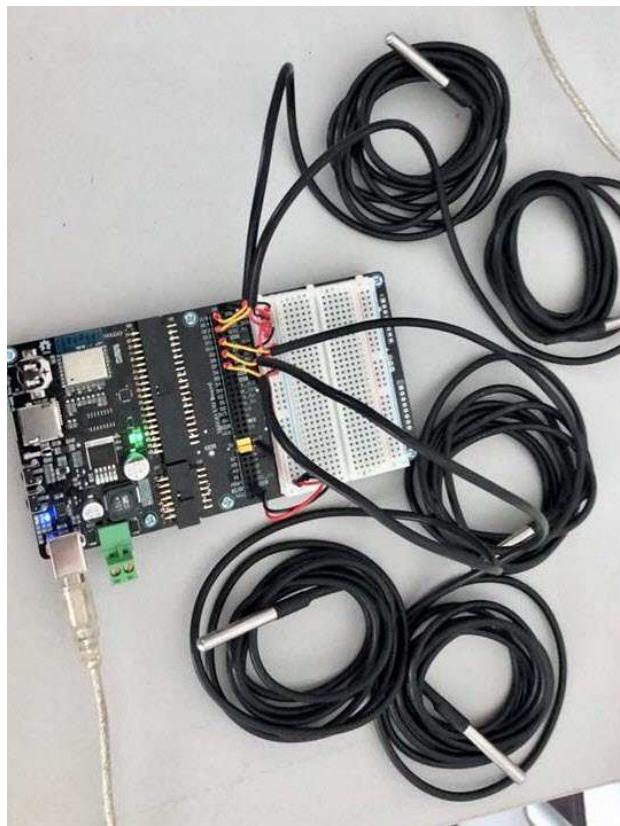
สำหรับบอร์ด MEGO ที่ขา Digital Pins : D0-D15 (16 Channels) มีความสามารถพิเศษรองรับ Digital Protocol ต่างๆ อาทิ OneWire, DHTxx และอื่นๆ

The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** DS18B20 | Arduino 1.8.4
- Menu Bar:** File Edit Sketch Tools Help
- Toolbar:** Includes icons for Open, Save, Upload, and Download.
- Sketch Area:** Displays the C++ code for the DS18B20 sketch. The code initializes five OneWire objects (T1-T5) and five DallasTemperature objects (Temp1-Temp5) corresponding to pins D7, D9, D11, D13, and D15. It then enters a loop where it requests temperatures from all sensors and prints the first one to the Serial Monitor.
- Status Bar:** Shows "Done compiling."
- Bottom Status:** Shows "1 Mego, 80 MHz, 4M (1M SPIFFS), v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM1"

ตัวอย่างนี้เป็นโปรแกรมว่านาค่า Temperature Sensors รุ่น DS18B20 ที่ใช้งานโปรโตคอล OneWire สามารถเรียกใช้งานได้ทุก Pins

หลักการทำงานของโปรแกรมจะทำงานแบบลำดับ โดยว่า Sensor 1 เรียงไปจนถึง Sensors ตัวสุดท้าย) ในตัวอย่างจะว่านาค่าอุณหภูมิ 5 Channels ที่ต่ออยู่ที่ Pin D7,D9,D11,D13,D15 แสดงผลที่ Serial Port โดยให้เปิด Serial Monitor ดูผลการทำงาน



ກາພຕັວອຍ່າງການອ່ານເໜ້ນເຂອງອຸນຫຼາມ DS18B20 ຈຳນວນ 5 Channels ທີ່ຕ່ວອຍູ່ກໍ Pin D7,D9,D11,D13,D15

```
#include <mega.h>
#include <OneWire.h>
#include <DallasTemperature.h>

OneWire * T1 = new OneWire(D7);
DallasTemperature * Temp1 = new DallasTemperature(T1);
OneWire * T2 = new OneWire(D9);
DallasTemperature * Temp2 = new DallasTemperature(T2);
OneWire * T3 = new OneWire(D11);
DallasTemperature * Temp3 = new DallasTemperature(T3);
OneWire * T4 = new OneWire(D13);
DallasTemperature * Temp4 = new DallasTemperature(T4);
OneWire * T5 = new OneWire(D15);
DallasTemperature * Temp5 = new DallasTemperature(T5);
```

```
void setup() {
Serial.begin(9600);
Temp1->begin();
Temp2->begin();
Temp3->begin();
Temp4->begin();
Temp5->begin();
}

void loop() {
float T;
Temp1->requestTemperatures();
T = Temp1->getTempCByIndex(0);
Serial.print("Temperature Sensor 1: ");
Serial.println(T);

Temp2->requestTemperatures();
T = Temp2->getTempCByIndex(0);
Serial.print("Temperature Sensor 2: ");
Serial.println(T);

Temp3->requestTemperatures();
T = Temp3->getTempCByIndex(0);
Serial.print("Temperature Sensor 3: ");
Serial.println(T);

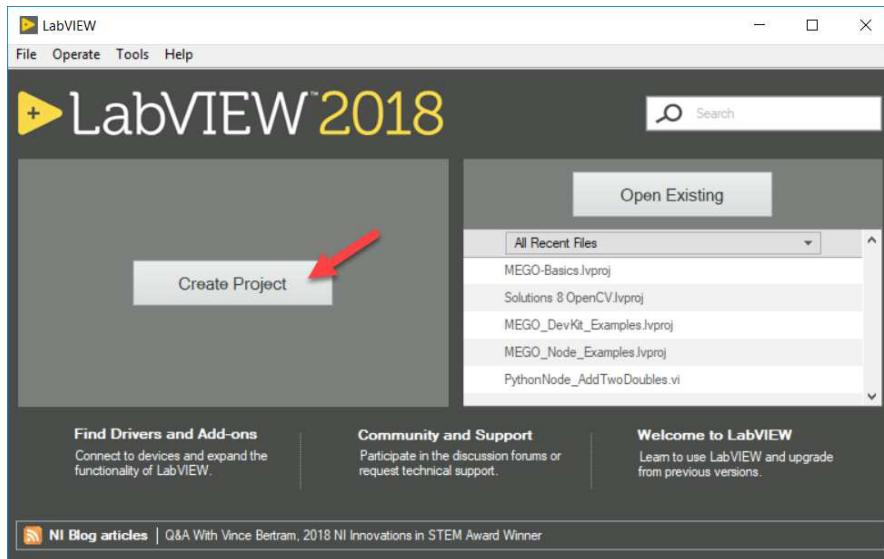
Temp4->requestTemperatures();
T = Temp4->getTempCByIndex(0);
Serial.print("Temperature Sensor 4: ");
Serial.println(T);

Temp5->requestTemperatures();
T = Temp5->getTempCByIndex(0);
Serial.print("Temperature Sensor 5: ");
Serial.println(T);

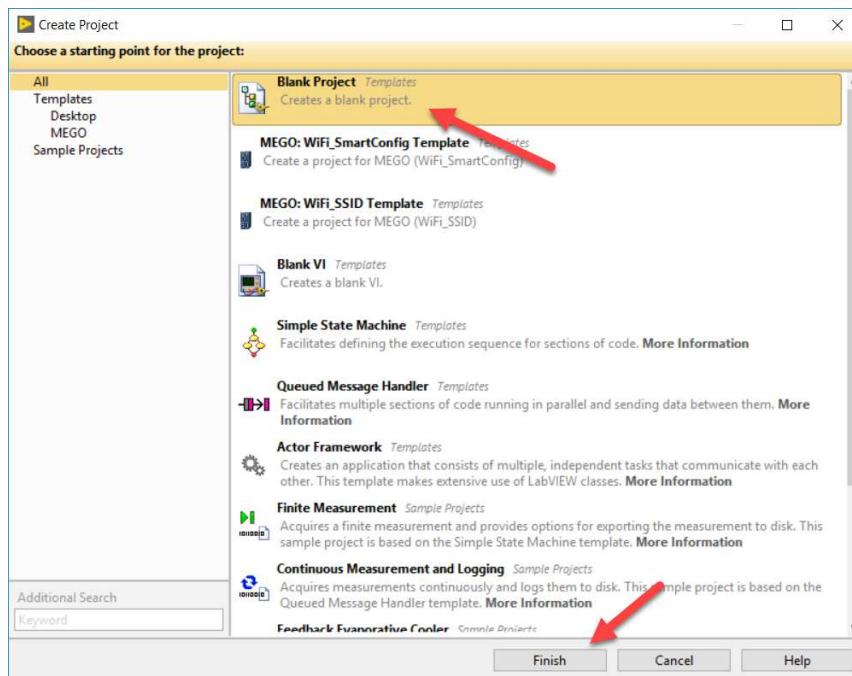
delay(500);
}
```

## เริ่มต้นเขียนโปรแกรม LabVIEW กับบอร์ด MEGO

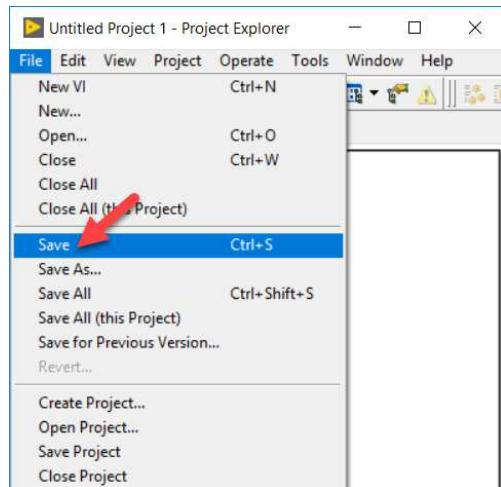
เริ่มต้นเปิดโปรแกรม LabVIEW จากนั้นจะขึ้นหน้าต่างดังรูป ให้เลือก Create Project เพื่อเริ่มต้นสร้างโปรเจคเขียนโปรแกรมกับบอร์ด MEGO



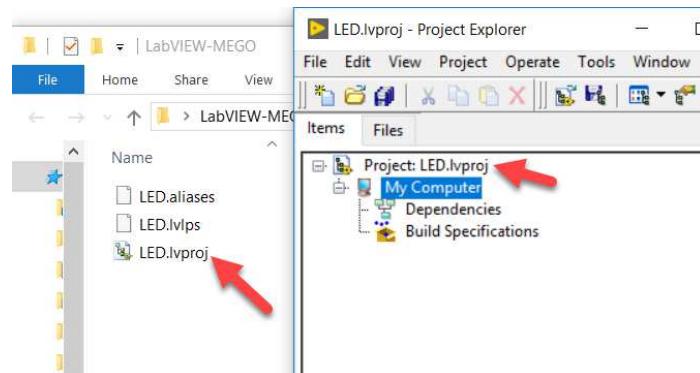
หน้าต่าง Create Project จะแสดงขึ้นมา เลือก Blank Project จากนั้นคลิก Finish



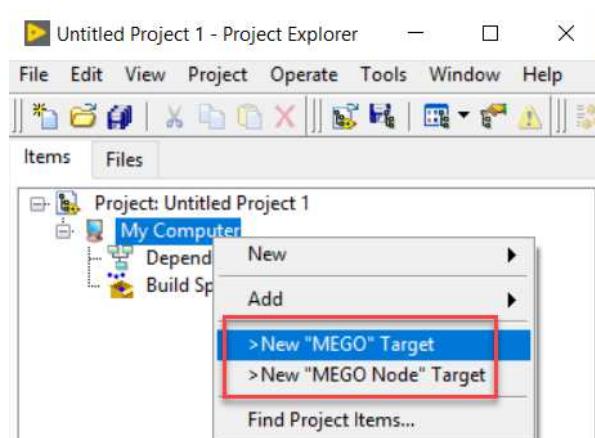
โปรเจคจะสร้างขึ้นมาและดังรูป ให้เลือก File > Save เพื่อตั้งชื่อโปรเจค ในกรณีนี้จะสร้างชื่อว่า LED โดยไฟล์โปรเจคจะมีนามสกุล \*.lvproj ดังรูป



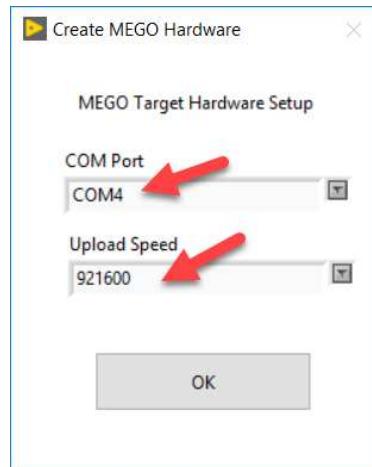
หลังจากการ Save จะมีไฟล์ จำนวน 3 ไฟล์ถูกสร้างขึ้นมา แสดงดังรูป โดยไฟล์ที่สำคัญคือ LED.lvproj สำหรับอีก 2 ไฟล์เป็นไฟล์ที่เก็บคุณลักษณะของโปรเจค ซึ่งไม่มีผลกับการเขียนโปรแกรมอย่างใด



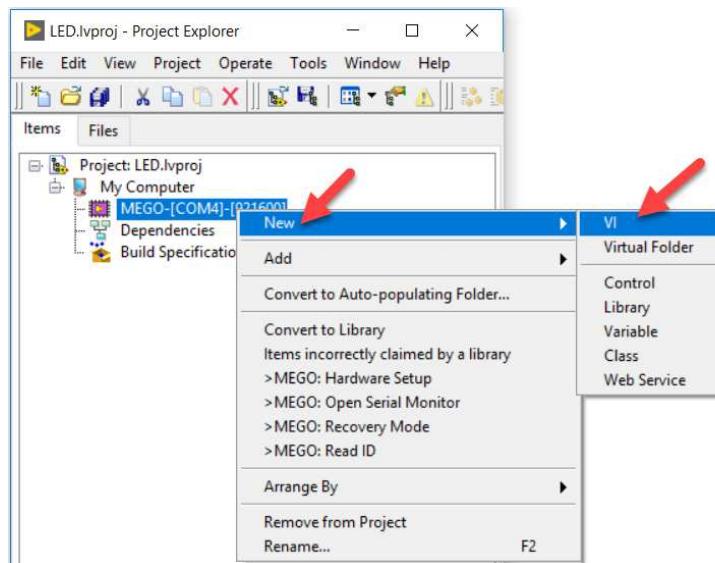
ในหน้าต่างโปรเจค ให้คลิกขวาที่ My Computer จากนั้นเลือกสร้างบอร์ด MEGO หรือ MEGO Node เข้ามาในโปรเจค ॥แสดงดังรูป



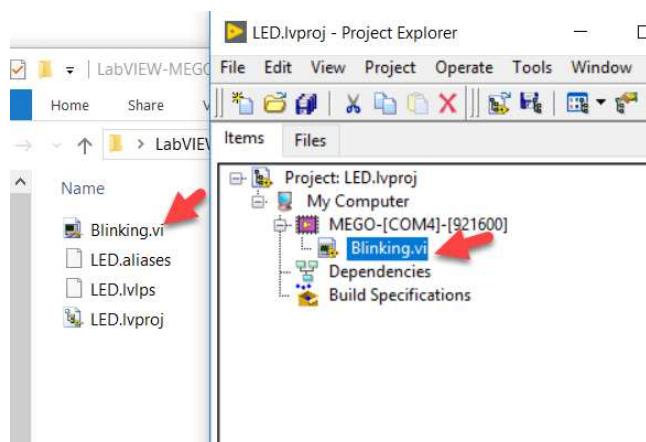
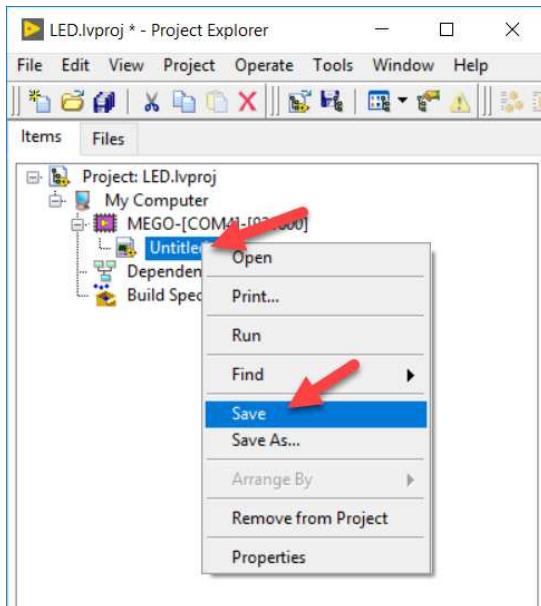
ให้เชื่อมต่อบอร์ด MEGO ผ่านสาย USB เข้ากับคอมพิวเตอร์ จากนั้นเลือก COM Port และความเร็วในการดาวน์โหลดไฟล์ที่ต้องการ ในกรณีที่ต้องเลือก Speed สูงสุดที่ 921600 Buad



จะเห็นบอร์ด MEGO ถูกสร้างเข้ามาในโปรเจคเรียบร้อย จากนั้นจะเป็นการสร้างโปรแกรม VI โดยคลิกขวาที่ บอร์ด MEGO เลือก New > VI ดังรูป



คลิก Save เพื่อบันทึกไฟล์โปรแกรม VI ตามชื่อที่ต้องการ ในกรณีเราทดลอง Save ชื่อว่า Blinking.vi เมื่อตรวจสอบใน Folder ที่เก็บไฟล์ ก็จะพบไฟล์ทั้งหมดดังรูป

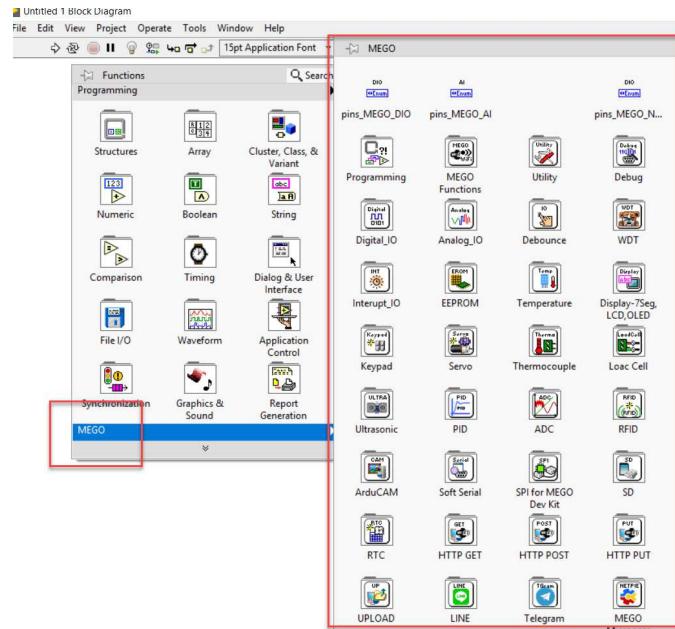


ดับเบิลคลิกที่ Blinking.vi จากนั้นจะมี 2 หน้าต่างเปิดขึ้นมา โดยหน้าต่างสีเทา เรียกว่า Front Panel โดยจะเป็นหน้าต่างในส่วนติดต่อผู้ใช้งาน User Interface ในกรณีที่เราพัฒนาโปรแกรมกับบอร์ด MEGO เราจะไม่ใช้หน้าต่าง Front Panel เมื่อจากว่าเราไม่สามารถสร้าง User Interface สำหรับบอร์ด MEGO ได้

ดังนั้นให้เข้าไปที่หน้าต่าง Block Diagram ซึ่งเป็นหน้าต่างเขียนโค้ด โดยสามารถใช้ Shortcut คือ Ctrl+E เพื่อสลับหน้าต่าง Front Panel กับ หน้าต่าง Block Diagram ได้

จากนั้นในพื้นที่ว่าง ให้คลิกขวา จะแสดงหน้าต่าง Function Palette ขึ้นมา ให้สังเกตว่า ในส่วนของเมนู Programming จะเป็นส่วนฟังก์ชันที่ทำงานบนคอมพิวเตอร์เท่านั้น ไม่รองรับการดาวน์โหลดลงบอร์ด MEGO

โดยฟังก์ชันที่สามารถดาวน์โหลดลงบอร์ด MEGO ได้นั้นจะอยู่ที่ MEGO โดยจะมีเมนูย่อย ต่าง ๆ มากน้อย แสดงดังรูป

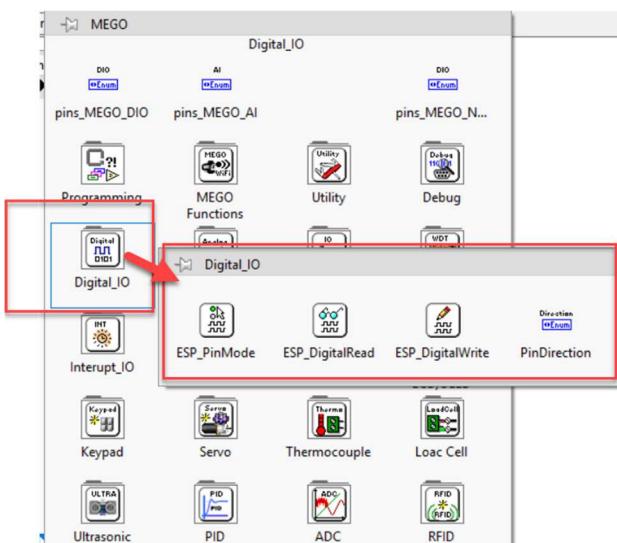


ณ ตอนนี้เราจะทดลองเขียนโปรแกรม ไฟกระพริบ Blinking LED โดยฟังก์ชันที่เราจะเรียกใช้ จะอยู่ที่เมนู MEGO > Digital IO จากนั้นให้หยิบฟังก์ชันดังต่อไปนี้มาวางที่ Block Diagram และดังรูป

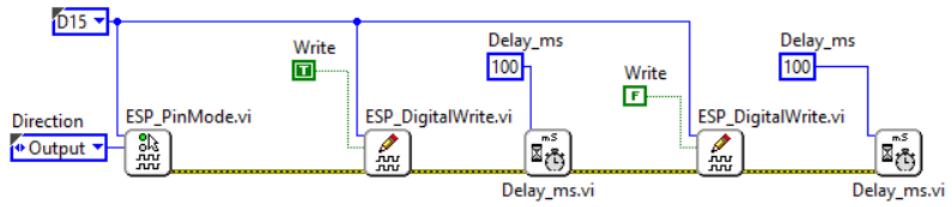
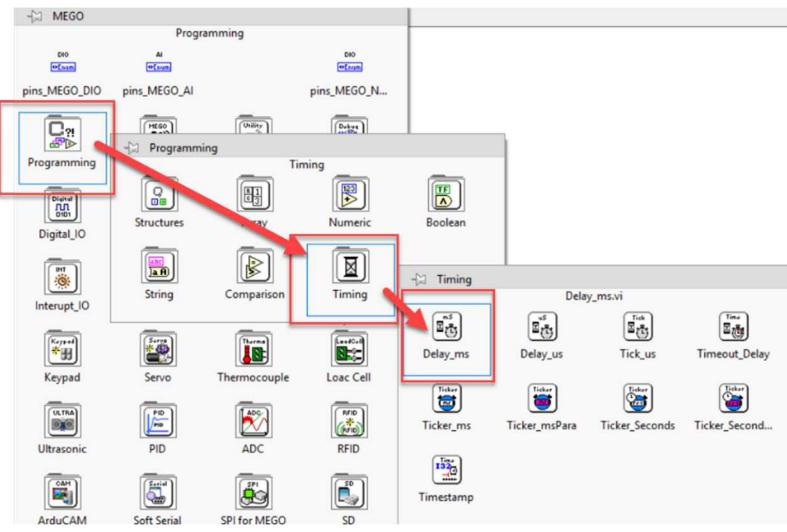
ESP\_PinMode.vi

ESP\_DigitalWrite.vi

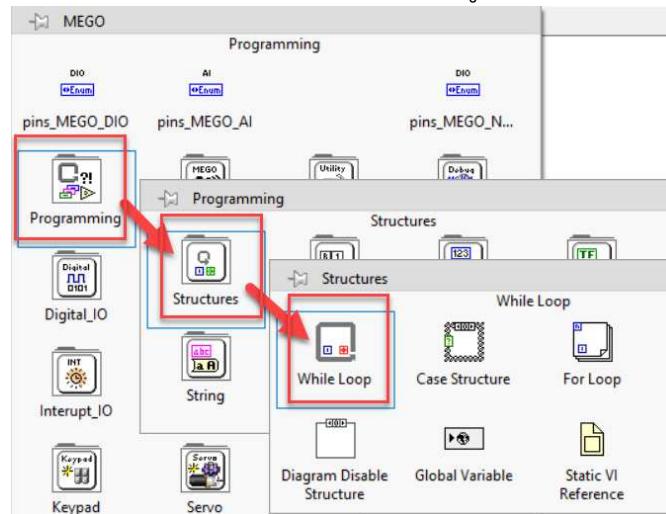
PinDirection.ctl



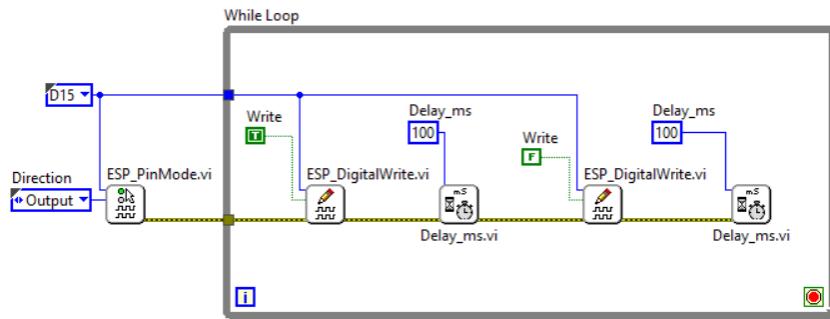
จากนั้นต้องใช้ฟังก์ชันที่ชื่อว่า Delay\_ms.vi เพิ่มเติม โดยจะอยู่ที่ MEGO > Programming > Timing > Delay\_ms.vi และดังรูป



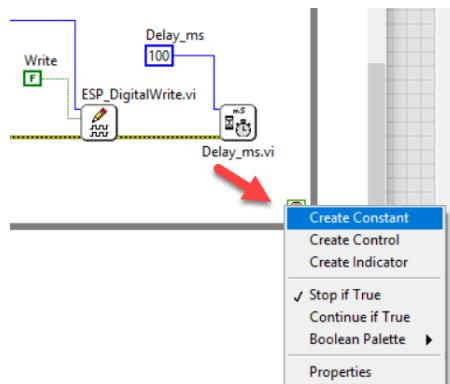
ขั้นตอนต่อไปเราต้องสร้างการวนลูป ของการกระแสเพื่อบอก LED เราจะเรียกใช้ While Loop พึ่งก์ซึ่น โดยหยิบได้จาก MEGO > Programming > Structure ดังรูป



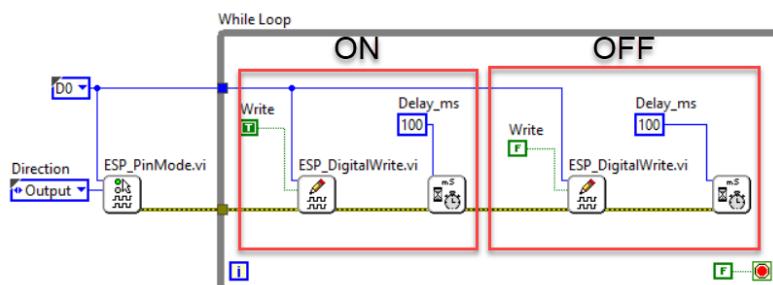
จากนั้นให้ลาก While Loop ครอบในส่วนของโค๊ดที่ต้องการ ดังรูป



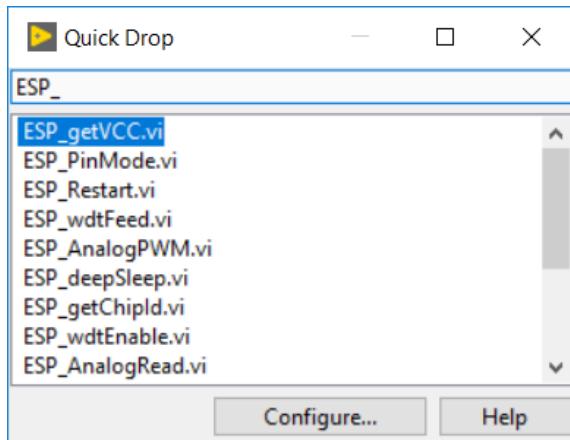
ขั้นตอนสุดท้าย คือ การใช้งาน While Loop เราต้องกำหนด เส้นไขในการหยุดลูปด้วย โดยจะกำหนดให้มาจาก Conditional Terminal ที่มุบล่างขวาของ While Loop โดยให้นำมาสู่ปีว่างที่จุดดังรูป จากนั้นคลิกขวา เลือก Create > Constant กำหนดให้เป็น F (False) แสดงว่าโปรแกรมนี้จะวนลูปไปตลอดเวลา โดยจะไม่มีการหยุดการทำงาน



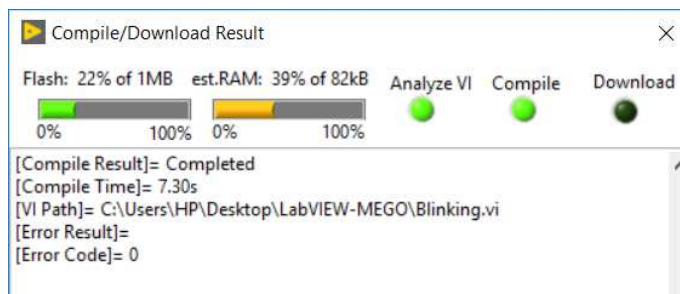
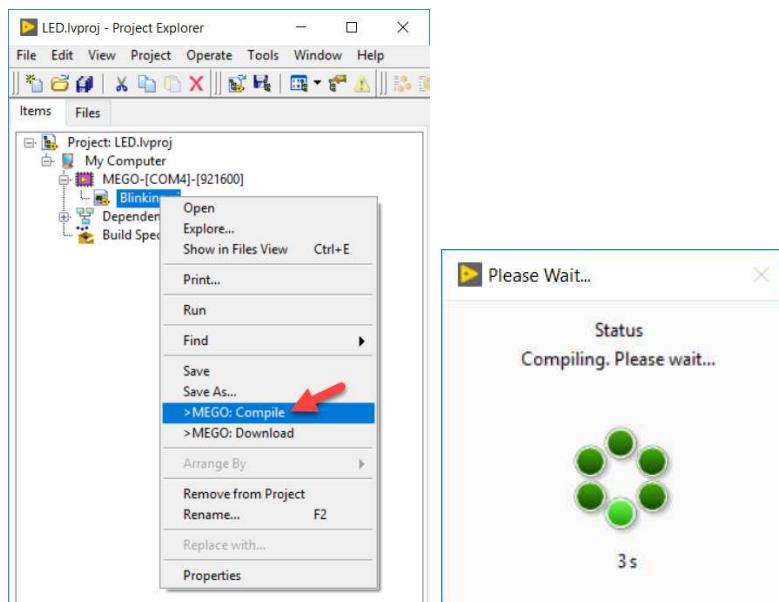
โดยการทำงานของโปรแกรม จะสั่งไฟ LED ติด และดับ ตามเวลา Delay ที่กำหนด จากนั้นจะวนลูปไปเรื่อยๆ ทำให้เราเห็นเป็นไฟกระพริบ



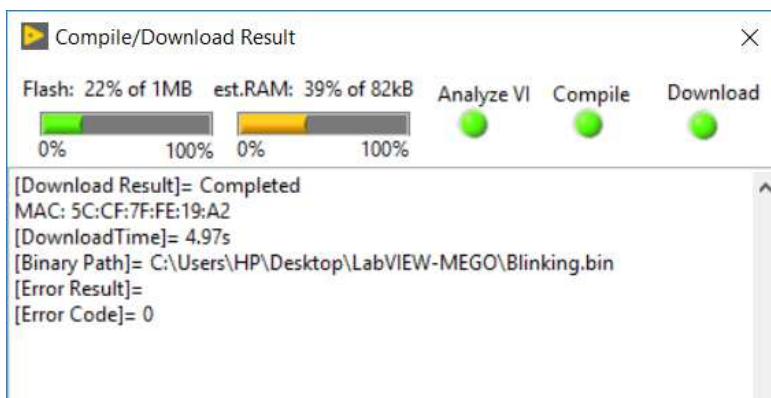
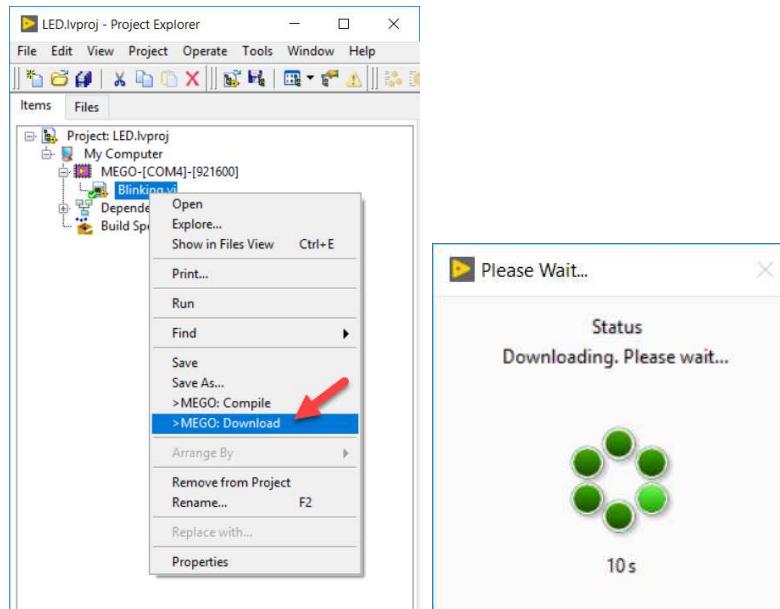
ในกรณีที่เราต้องการเพิ่มเติมฟังก์ชันเข้ามาในโปรแกรม เราสามารถใช้คีย์ลัด Quick Drop ได้ โดยการกด Ctrl + Spacebar จากนั้นพิมพ์ชื่อฟังก์ชันที่ต้องการ เราจะสามารถเขียนโปรแกรมได้ง่าย โดยไม่ต้องคลิกขวา แล้วหยิบฟังก์ชันต่างๆ มาจาก Function Palette ซึ่งวิธีนี้เป็นที่นิยมใช้งาน เพราะเพียงแค่รูซื้อฟังก์ชันก็สามารถนำไปใช้ง่าย และรวดเร็ว



กลับมาที่ Project ให้ทำการ Save โปรแกรมให้เรียบร้อย จากนั้นคลิกขวาที่ไฟล์ Blinking.vi เลือก > Compile จะปรากฏหน้าต่าง Compiling เมื่อเสร็จแล้วจะแสดงหน้าต่าง ผลการ Compile ดังรูป

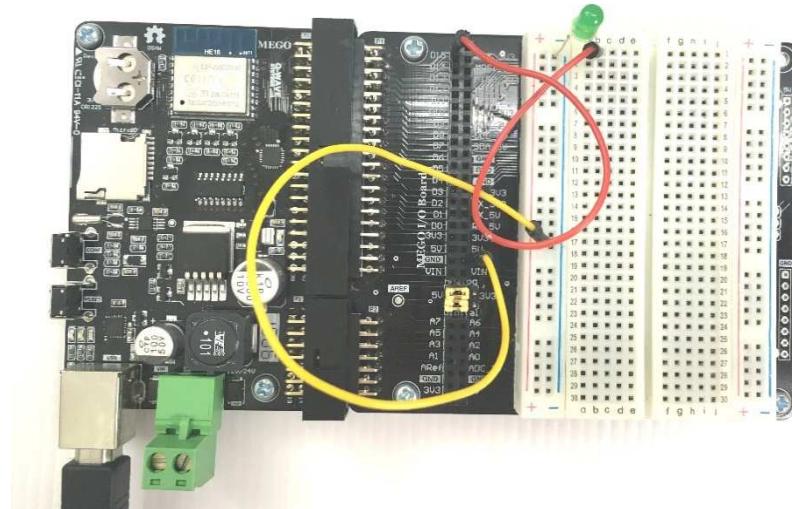


กลับมาที่ Project ให้คลิกขวาที่ Blinking.vi เลือก Download รอสักครู่ โปรแกรมจะทำการดาวน์โหลดคิ้ดลงบอร์ด MEGO โดยระยะเวลาในการโปรแกรมในขั้นตอนนี้จะขึ้นอยู่กับการเลือก Download Speed โดยสามารถเลือกความเร็วสูงสุดได้ที่ 921600 Buad ซึ่งจะใช้เวลาประมาณ 5-10 วินาที

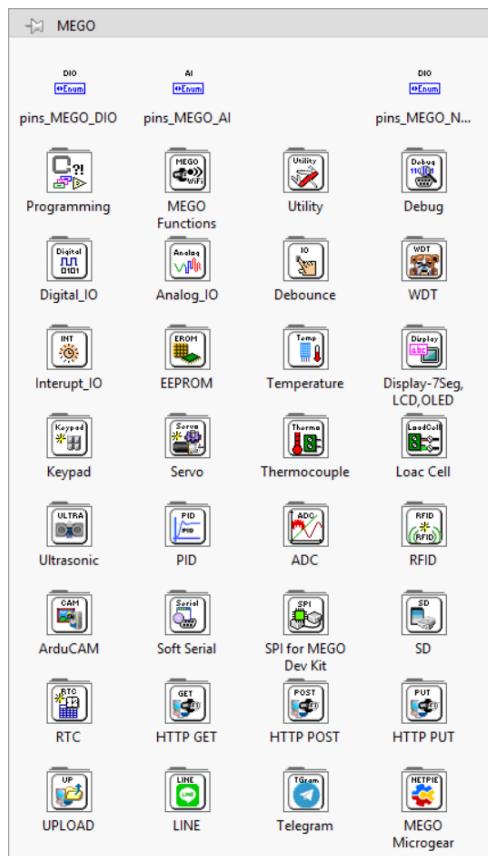


ถึงขั้นตอนนี้ให้ดูผลของไฟกระพริบที่บอร์ด MEGO ถ้าทำงานได้ตามที่เขียนโปรแกรมแสดงว่า ขั้นตอนถูกต้อง จากนั้นสามารถกลับไปที่โปรแกรมเพื่อแก้ไขค่า Delay (ms) จากนั้น Compile/Download อีกรอบ เพื่อดูผลการเปลี่ยนแปลง

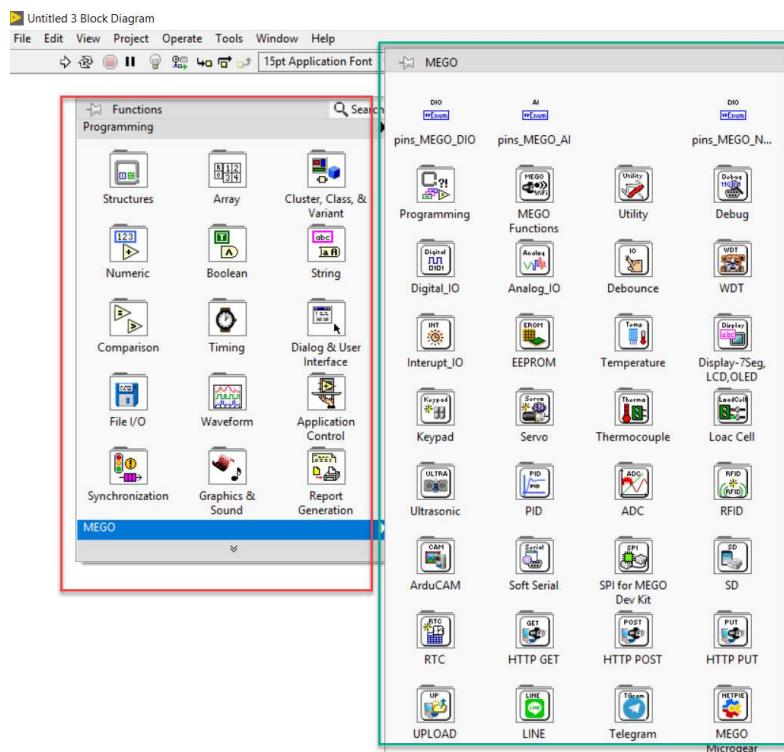
\*ทุกครั้งก่อนที่จะ Compile ให้ทำการ Save VI ทุกครั้ง ไม่เช่นนั้น โปรแกรมจะนำเอาโค้ดเดิมไป Compile



## MEGO Generic Functions



ฟังก์ชันพื้นฐานที่ใช้งานกับบอร์ด MEGO จะอยู่ที่ Palette MEGO เท่านั้น ดังรูป โดยฟังก์ชันอื่น ๆ จะไม่รองรับ ซึ่งการเรียกใช้งานฟังก์ชันที่ไม่รองรับจะทำให้ Compile ไม่ผ่าน และไม่สามารถโปรแกรมลงบอร์ด MEGO ได้



โดยพังก์ซึ่งรับแบ่ง ๆ แบ่งหมวดหมู่ ประกอบไปด้วย

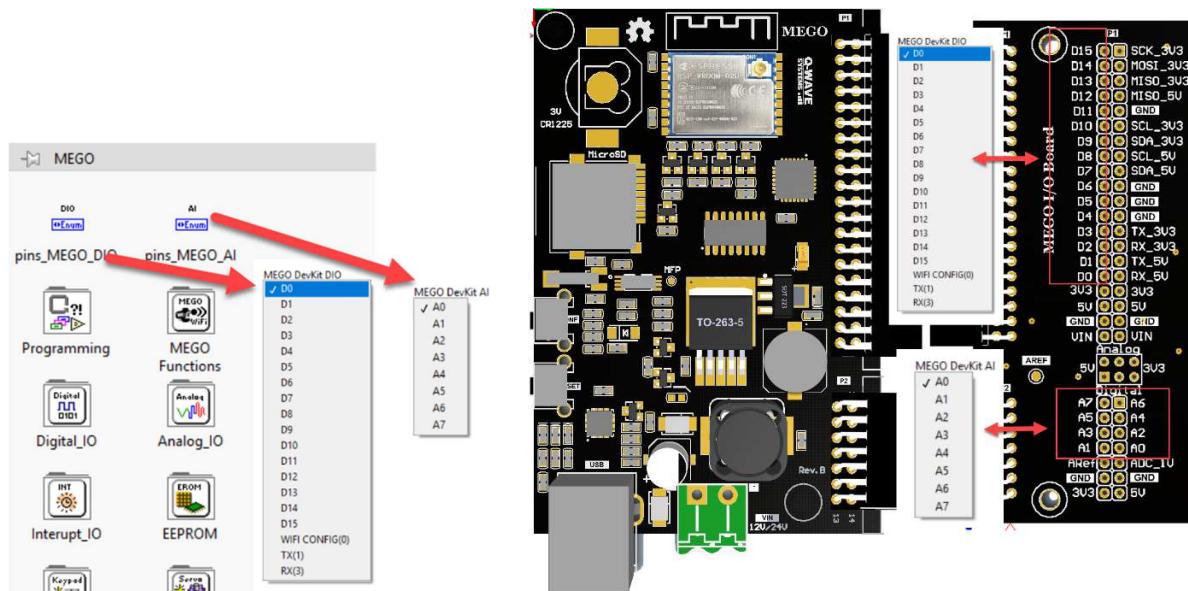
ສໍາຮຽບບວርດ MEGO

Pin\_MEGO\_DIO – เป็นการกำหนดขา Digital ที่ต้องการใช้งาน โดยจะตรงกับชาร์ดแวร์ D0-D15

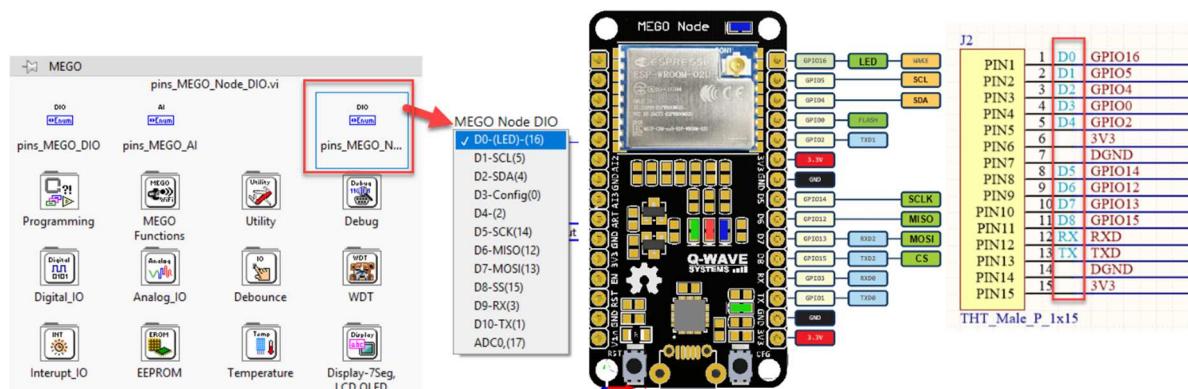
Pin\_MEGO\_AI - เป็นการกำหนดขา Analog ที่ต้องการใช้งาน โดยจะตรงกับชาร์ดแวร์ A0-A7

## ສໍາຮັບບວດ MEGO Node

Pin\_MEGO\_Node\_DIO – เป็นการกำหนดขา Digital D0-D10

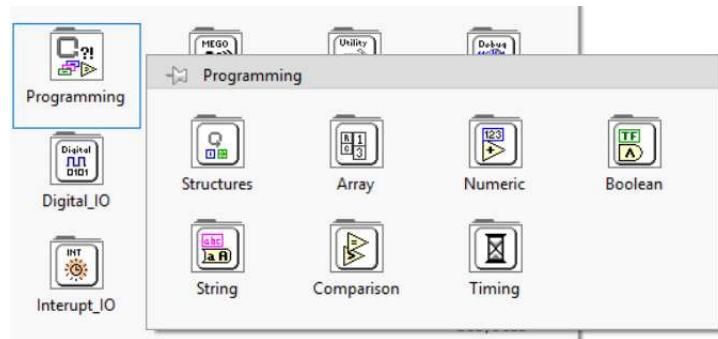


## การกำหนดข้าบอร์ด MEGO



## การกำหนดข้าบอร์ด MEGO Node

Programming



Structure เป็นฟังก์ชันคำสั่งพื้นฐานในการเขียนโปรแกรม ซึ่งกำหนดการทำงานของโปรแกรมทั้งหมด

Array เป็นฟังก์ชันที่เกี่ยวกับการตัวแปรของข้อมูลชนิด Array (อะเรย์)

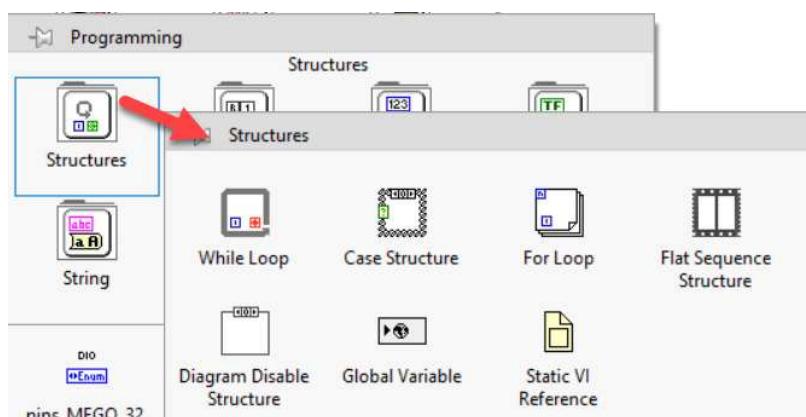
Numeric เป็นฟังก์ชันที่เกี่ยวกับการตัวแปรของข้อมูลชนิด Numeric (ตัวเลข)

String เป็นฟังก์ชันที่เกี่ยวกับการตัวแปรของข้อมูลชนิด Array (ตัวอักษร)

Comparison เป็นฟังก์ชันที่เกี่ยวกับการคำนวณ และการเปรียบเทียบ

Timing เป็นฟังก์ชันที่เกี่ยวกับการจัดการด้านเวลา Timing

## Structure



While Loop เป็นฟังก์ชันกำหนดวนรอบการทำงานของโปรแกรม ซึ่งทุกโปรแกรมจะมี While Loop เสมอ

Case Structure เป็นฟังก์ชันเลือกตัดสินใจการทำงานตามเงื่อนไข หรือชนิดของข้อมูลที่เข้ามา โดยการทำงานคล้ายกับ If then else หรือ switch case กันไป

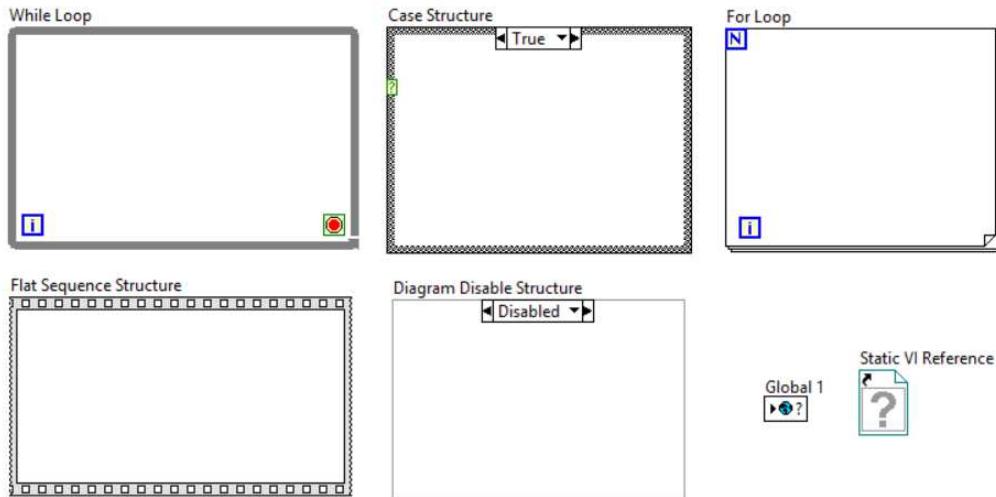
For Loop เป็นฟังก์ชันวนรอบการทำงานแบบกำหนดได้ โดยค่า N เป็นตัวกำหนดจำนวนรอบการทำงาน

Flat Sequence Structure สามารถปรับแต่งให้เป็น Stacked Sequence Structure ได้ โดยเป็นฟังก์ชันกำหนดกรอบการทำงานของโปรแกรมในส่วนย่อย ๆ โดยบักใช้เพื่อกำหนดลำดับการทำงาน หรือจัดระเบียบของโปรแกรมให้เข้าใจง่าย

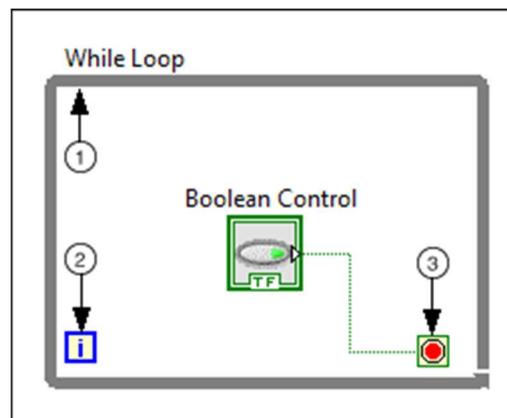
Diagram Disable Structure เป็นฟังก์ชันปิดการทำงานของโปรแกรมบางส่วน เสมือนคำสั่ง // Comment ในภาษาอื่น โดยสามารถเปิด-ปิด การทำงานของโคลด์ในส่วนใดก็ได้

Global Variable เป็นการสร้างตัวแปรชนิด Global โดยสามารถเรียกใช้จากโค๊ดในส่วนใดก็ได้ของโปรแกรม ช่วยลดจำนวน Wire ที่เชื่อมต่อข้อมูล ทำให้โค๊ดเป็นระเบียบมากขึ้น

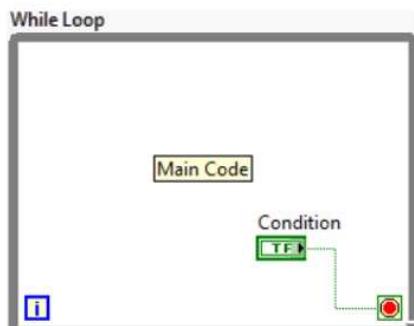
Static VI Reference เป็นการสร้าง Reference ของ VI โปรแกรมย่อย จากนั้นเรียกใช้ในโปรแกรมหลักได้อย่างสะดวก เสมือนเรียก Sub Function เพื่อใช้งาน



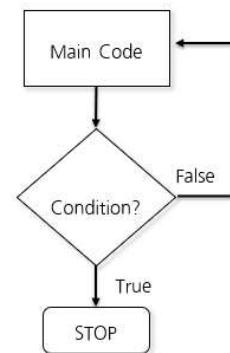
While Loop



พังก์ชัน While Loop เป็นพื้นฐานของโปรแกรม หลักการทำงานแสดงดัง Flow Chart ดังรูปโดยจะทำงานแบบวนลูปต่อเนื่อง ไปจนกว่าเงื่อนไขในการหยุดลูปจะเป็นจริง (True)



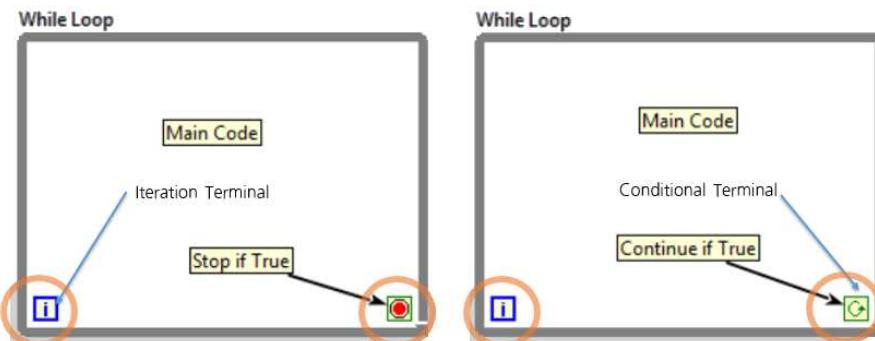
LabVIEW While Loop



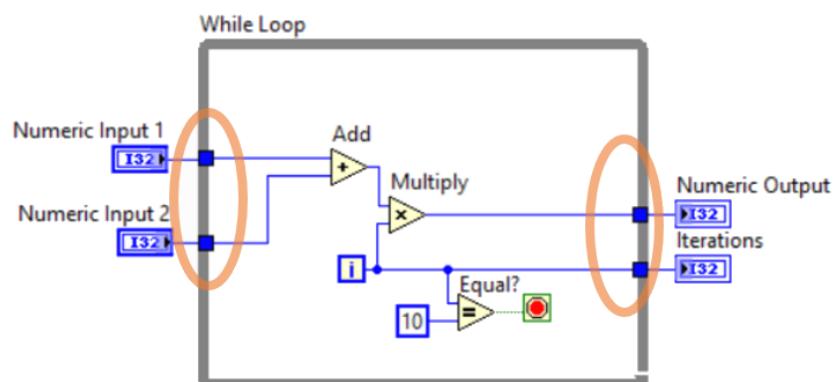
Flow Chart

Iteration Terminal = โดย i เป็นตัวบันทึกจำนวน Loop ที่ทำงาน โดยเพิ่มขึ้นทีละ 1 ค่าเริ่มต้นบันจาก 0 (Zero-Index)

Conditional Terminal = เป็นตัวกำหนดเงื่อนไขในการหยุด Loop โดยเมื่อเลือกใช้งาน 2 เงื่อนไข ดังนี้ Stop if True หรือ Continue if True

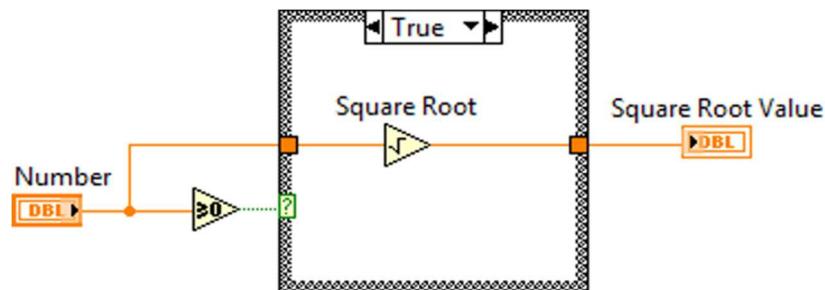


การใช้งาน While Loop มักจะใช้งานร่วมกับการส่งค่าผ่าน Loop โดยเรียกว่า Tunnel ซึ่ง ทางด้านซ้ายจะเป็นการส่งค่าเข้าใน Loop และด้านขวาจะเป็นการส่งค่าผ่านออก Loop

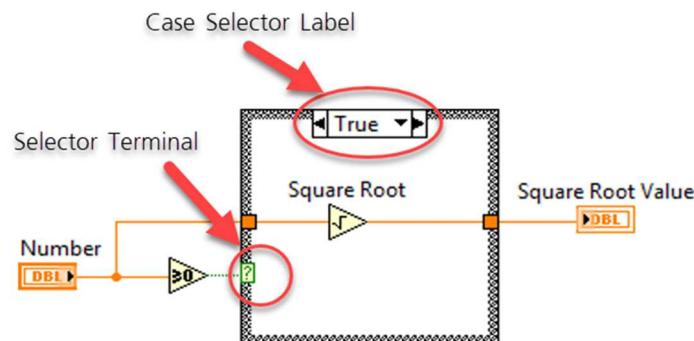


Tunnels เป็นจุดที่ส่งค่า In/Out ผ่าน While Loop โดยสามารถอยู่ในด้านใดก็ได้ของ While Loop (บิยมกำหนดด้านซ้ายเป็น Input และด้านขวาเป็น Output) While Loop จะเริ่มต้นทำงานก็ต่อเมื่อเมื่อ Input เข้ามาทั้งหมด While Loop จะวนทำงานภายในจนกว่า เงื่อนไขในการ STOP Loop เป็นจริง (True)

### Case Structure



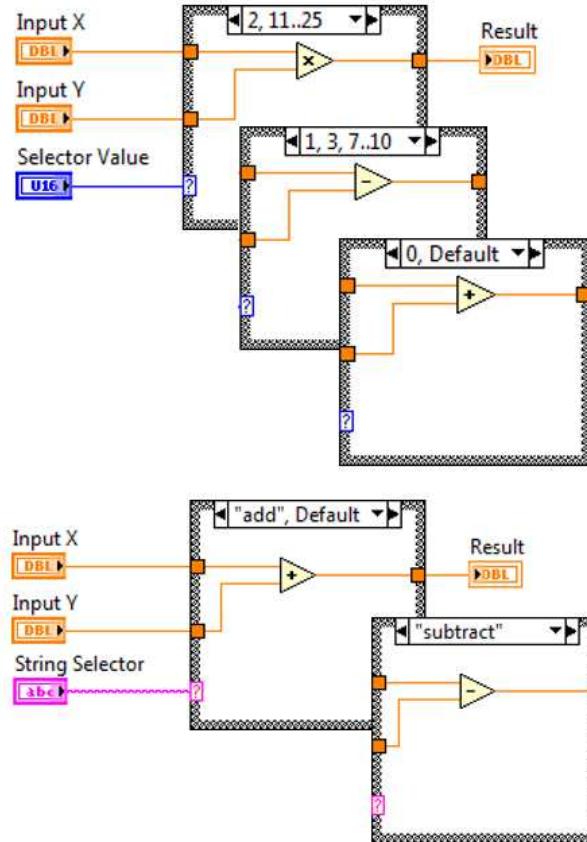
เป็นฟังก์ชันนิยมใช้งานเป็นการเลือกตัดสินใจการทำงานอย่างเดียว ตามเงื่อนไขที่กำหนด หรือคล้ายกับ switch case ก็ว่าไป



โดยจะต้องมี Case การทำงานอย่างน้อย 2 เคส ขึ้นไป โดย Selector Terminal จะเป็นตัวกำหนดว่า Case ไหนจะทำงาน Case จะทำงานตามเงื่อนไขที่เข้ามา โดยจะต้องเลือกการทำงานจากคลาสได้ เคสหนึ่งเท่านั้น เปรียบเทียบได้กับคำสั่ง if...then...else ในภาษา Text Programming

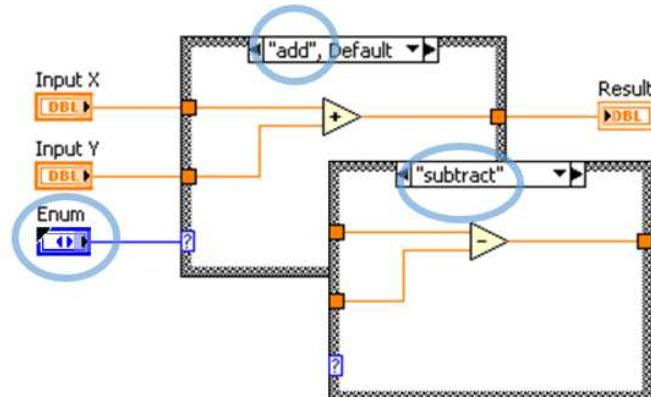
Case Selector Label จะแสดงชื่อของเคส ที่มีทั้งหมดที่มี  
Selector Terminal กำหนดเงื่อนไขว่าเคสใดจะทำงาน

การใช้งานชนิดของตัวแปร Selector terminal: ลักษณะการใช้งานเลือกจากชนิดของตัวแปรที่เข้ามา โดยจะต้องมี “Default Case” เสมอ



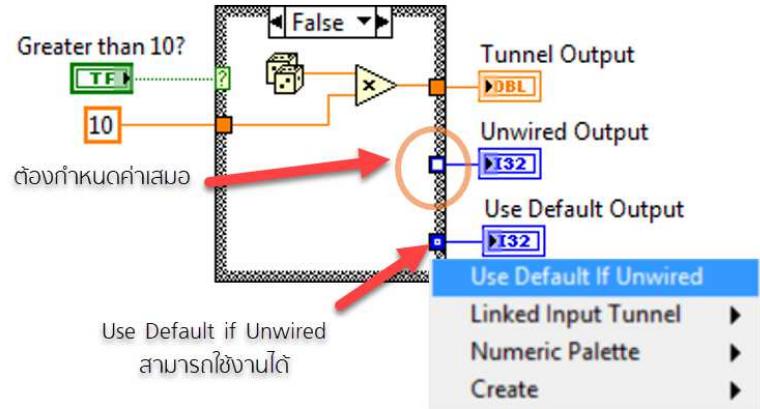
Boolean True case หรือ False Case

Integer หรือ Enum สามารถกำหนดค่า ตัวเลขได้ๆ ก็ได้ Enum ก็คือตัวเลขชนิดหนึ่งบันทุณเอง String สามารถมีค่า ได้ๆ ก็ได้ตามรายการความ ที่กำหนด



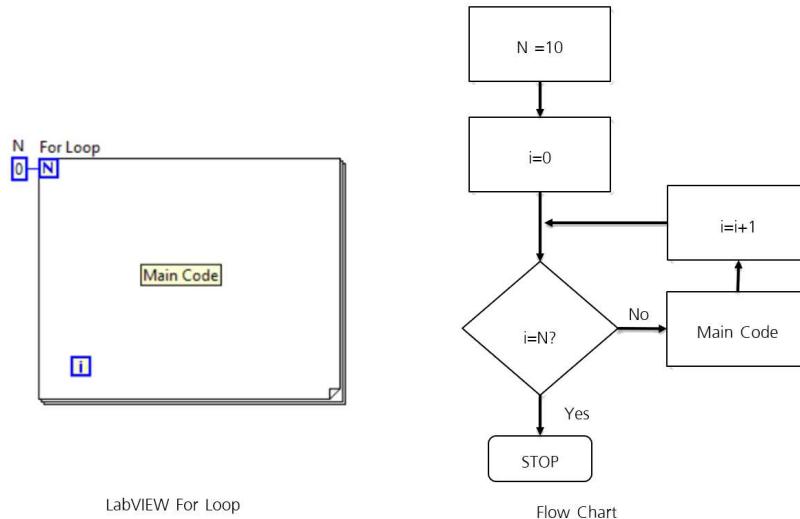
ตัวอย่างการใช้งาน Enum Case Structure (ตัวแปร Enum) โดย Enum นิยมใช้งานจำนวนมากในการใช้งานร่วมกับ Case Structure เพราะสามารถแสดงชื่อของค่าในลักษณะที่เข้าใจง่าย ถึงแม้ว่าข้อมูลในค่าจะเป็นตัวเลข Integer ก็ตาม

การใช้งาน Input/Output Tunnels สำหรับ Case Structure โดย Input Tunnels สามารถนำเข้าไปได้โดยไม่จำเป็นต้องเชื่อมต่อใช้งานทั้งหมด และ Output Tunnels มีเงื่อนไขว่า จะต้องเชื่อมต่อออกมากันหมด ทุก Case ไม่ เช่นนั้นจะ Run VI ไม่ได้

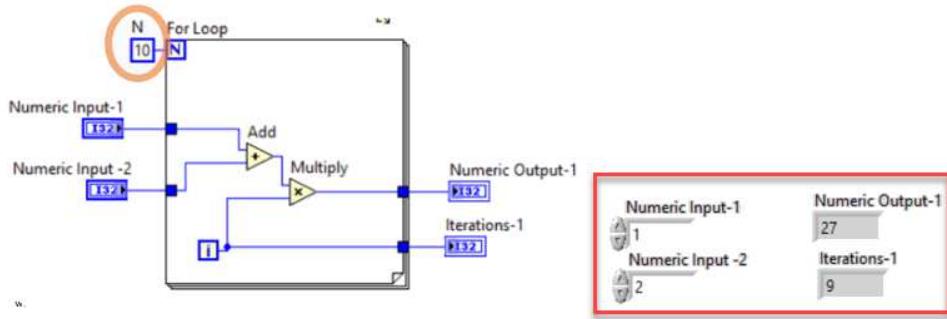


สามารถกำหนด “Use Default if Unwired” สำหรับค่าที่ไม่ต้องการเชื่อมต่อข้อมูล โดย คลิกขวาที่ Tunnels ที่ต้องการ > เลือก “Use Default if Unwire” ค่าเริ่มต้นของตัวแปรคือ Numeric =0 , Boolean = False, String = Empty

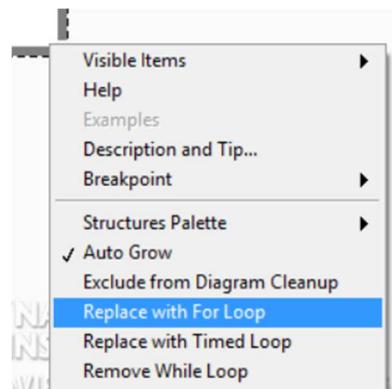
### For Loop



เป็นฟังก์ชันวนรอบการทำงานแบบกำหนดได้ โดยค่า N เป็นตัวกำหนดจำนวนรอบการทำงาน โดย Flow Chart การทำงานแสดงดังรูป โดยค่า N จะเริ่มต้นที่ 0 (Zero index) และจะเพิ่มขึ้นไปเรื่อย ๆ จนกว่า ค่า N จำนวนรอบจะครบตามกำหนด

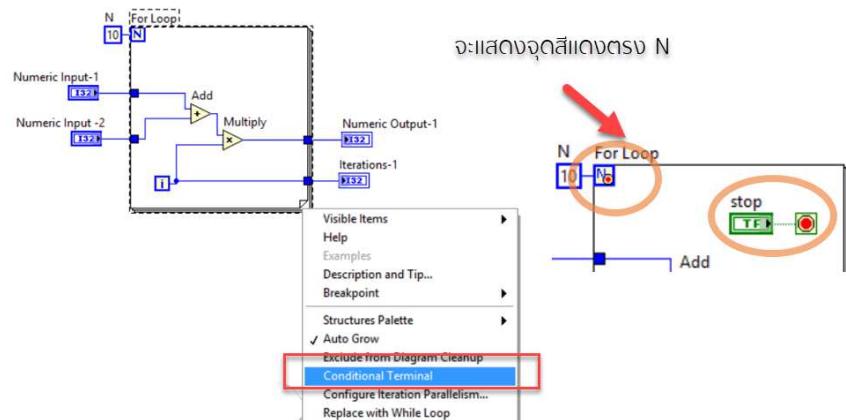


For Loop จะทำงานตามจำนวนที่กำหนดด้วย “N” เช่น กำหนดค่า =10 หมายถึง Loop จะทำงานเพียง 10 ครั้ง จากนั้นจะหยุดและออกจาก Loop โดยจากตัวอย่างโปรแกรม จะได้ผลค่า Output = 27 และค่า Iteration = 9

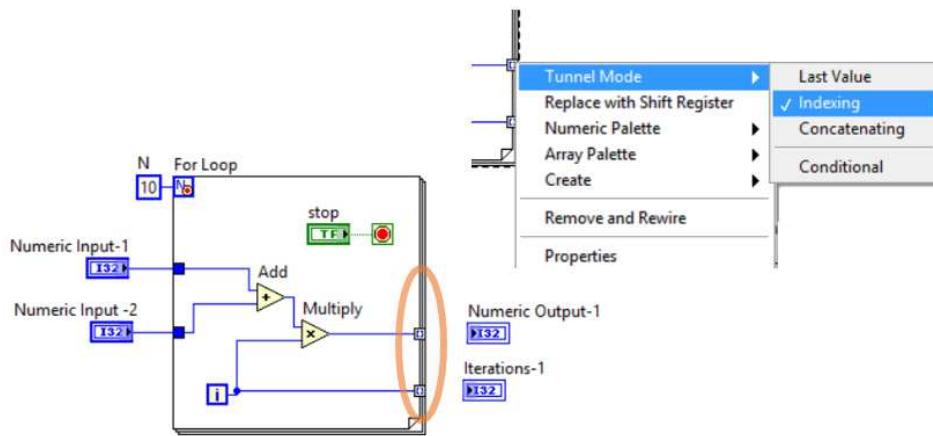


เราสามารถ Replace While Loop ด้วย For Loop ด้วยวิธี คลิกขวา > “Replace with For Loop” แสดงดังรูป

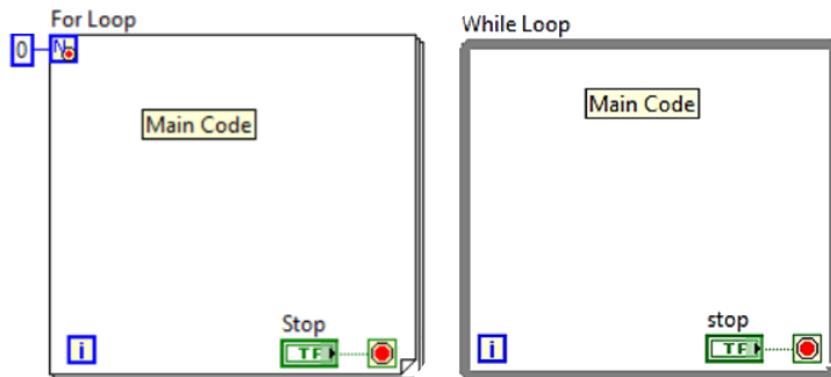
โดย For Loop สามารถกำหนดให้มี Conditional Terminal ได้เช่นกัน โดยคลิกขวา > เลือก Conditional Terminal



ค่าเริ่มต้นที่ Output Terminal ของ For Loop จะถูกกำหนดเป็น Array อัตโนมัติ (Indexing) ถ้าต้องการเปลี่ยน คลิกขวา > Tunnel Mode > ... โดยสามารถเลือก Last Value หรือ Concatenating

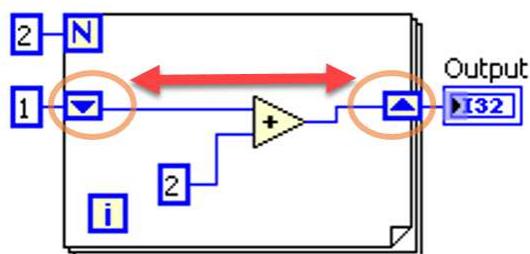


ความแตกต่างระหว่าง For Loop และ While Loop

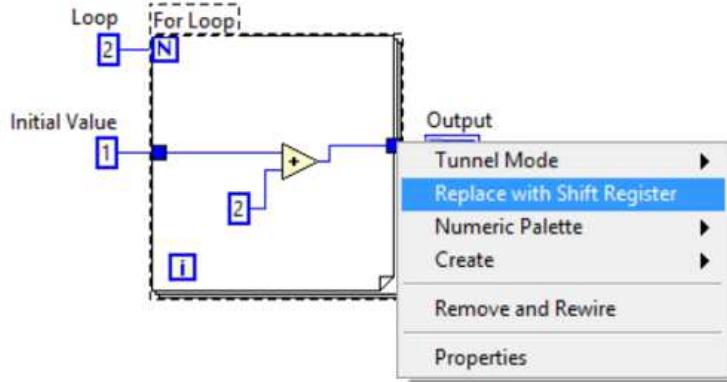


For Loop จะทำงานจนครบจำนวน N Loop แต่ถ้า Conditional Terminal เป็น True จะหยุด Loop ก่อน ถ้ากำหนด  $N = 0$  หมายความว่า For Loop จะไม่ทำงานเลยแม้แต่ครั้งเดียว และค่าเริ่มต้นของ Output Terminal จะเป็น Array อัตโนมัติ (Indexing)

While Loop จะหยุดการทำงานเมื่อ Conditional Terminal เป็น True โดย Loop จะทำงานอย่างน้อย 1 ครั้งเสมอ และค่าเริ่มต้นของ Output Terminal จะเป็น ค่าสุดท้ายก่อนออกจาก Loop (Last Value)

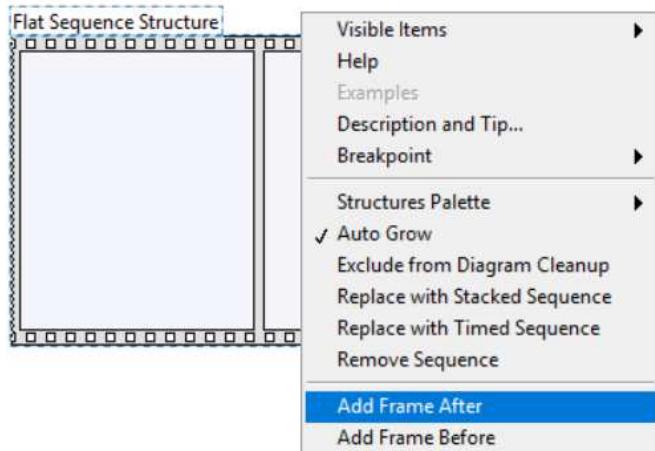


ในกรณีที่ต้องการส่งค่าระหว่าง Loop (Feedback in Loops) ต้องใช้ Shift Register และดังรูป การกำหนด Shift Register จะต้องกำหนด 2 จุด Input และ Output โดยคลิกขวาที่ Tunnel ได้ก็ได้ เลือก “Replace with Shift Register” จากนั้นคลิกที่ Shift Register วัดด้านหนึ่ง

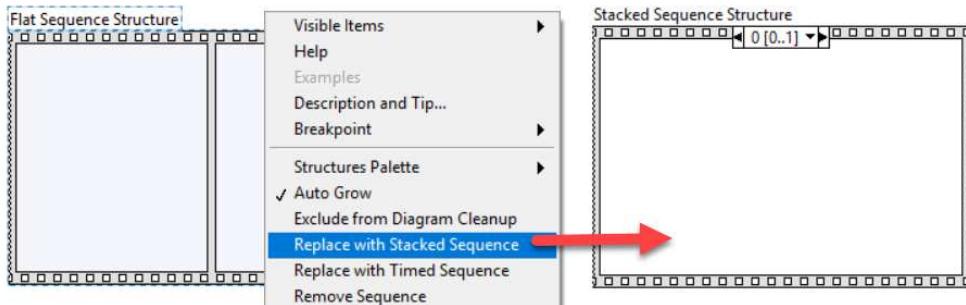


โดยด้านจุดด้าน Output จะเป็นค่าสุดท้ายก่อนออกจาก Loop และส่งต่อไปให้ Shift Register ด้าน Input จากนั้นจะเป็นการส่งค่ากลับไปให้ Loop กดไปเรื่อย ๆ จนกว่าจะออกจาก Loop (สามารถใช้งาน Shift Register ร่วมกับ While Loop ได้)

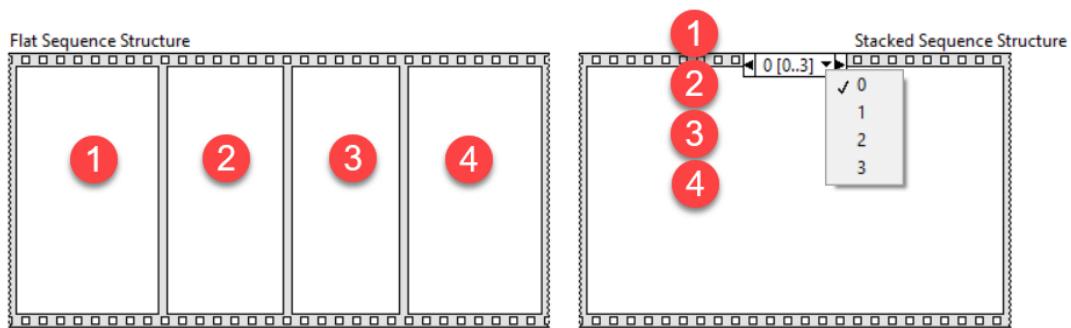
#### Flat Sequence Structure



เป็นการสร้าง Sequence เพื่อกำหนดลำดับการทำงานของโค้ด (Data Flow) โดยที่มีต้องเชื่อมต่อสาย (Wiring) สามารถเพิ่ม Sequence ได้ด้วย คลิกขวา > Add frame After

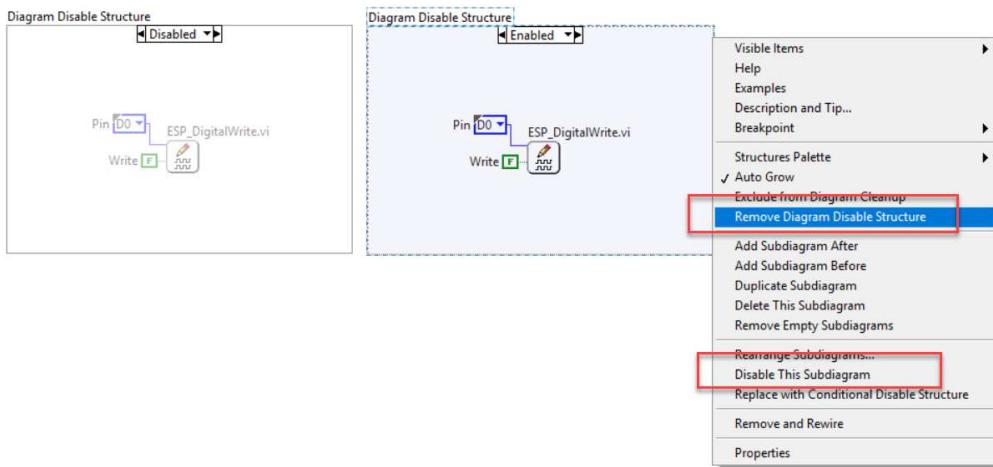


โดยเราสามารถเรียกใช้งาน Stacked Sequence structure ได้โดยการคลิกขวา > Replace with Stacked Sequence และดังรูป



การใช้งานจริงเราเนี่ยมเรียกใช้ Stacked Sequence structure เมื่อจากว่าโค๊ดในแต่ละส่วนจะอยู่ภายใน Sequence ซึ่งจะทำให้โค๊ดดูเป็นระเบียบ อ่านง่าย และเขียนต่ออยอดเพิ่มเติมได้สะดวก

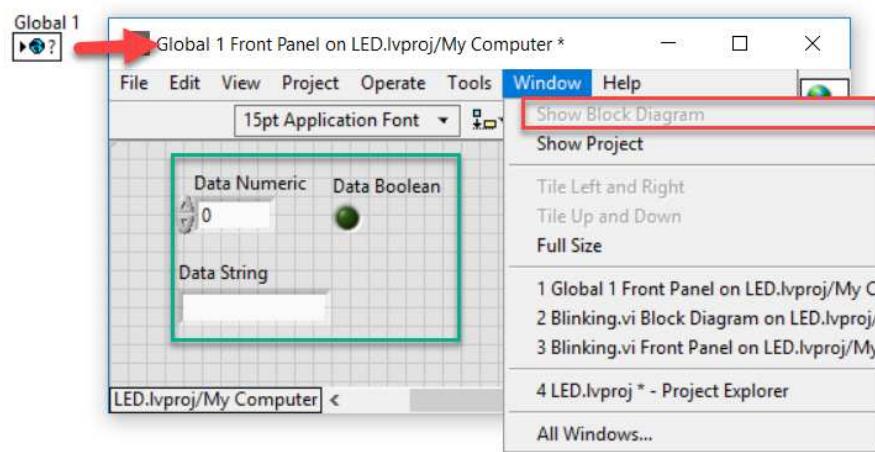
### Diagram Disable Structure



เป็นฟังก์ชันปิดการทำงานของโปรแกรมบางส่วน เสมือนคำสั่ง // Comment ในภาษาอื่น โดยสามารถเปิด-ปิด การทำงานของโค๊ดในส่วนใดก็ได้ โดยสามารถ Enable หรือ Disable ได้ด้วยวิธีการคลิกขวา หรือถ้าต้องการลบ ก็ให้เลือก Remove Diagram Disable Structure

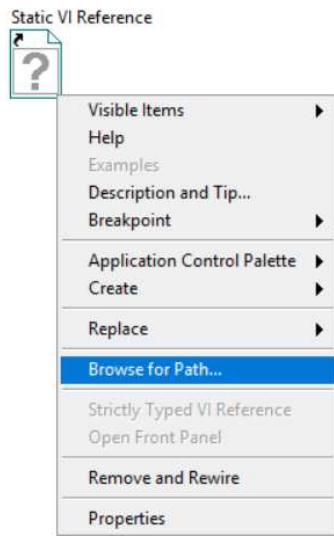


## Global Variable



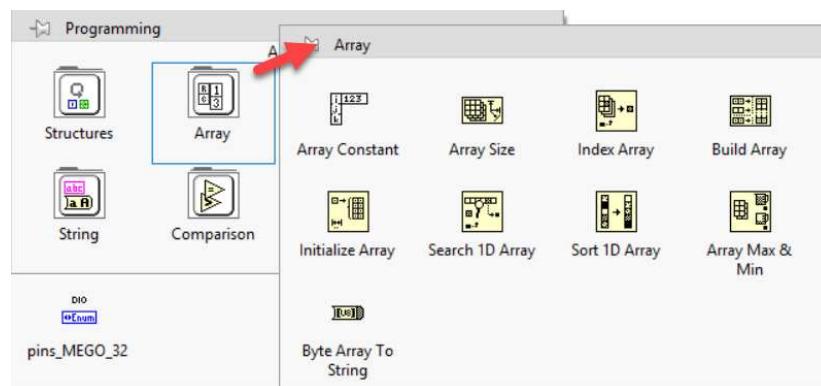
เป็นการสร้างที่ตัวแปรชุด Global โดยจะเป็นการสร้าง New VI ขึ้นมาใหม่ จากนั้นต้อง Save ตั้งชื่อเหมือน VI ปกติ โดยที่ไฟล์นี้จะมีนามสกุล .vi แต่จะไม่มี Block Diagram จะมีแต่ส่วนของ Front Panel ก็จะเก็บตัวแปรต่าง ๆ ก็สร้างขึ้นมาเพื่อใช้งานเท่านั้น

## Static VI Reference

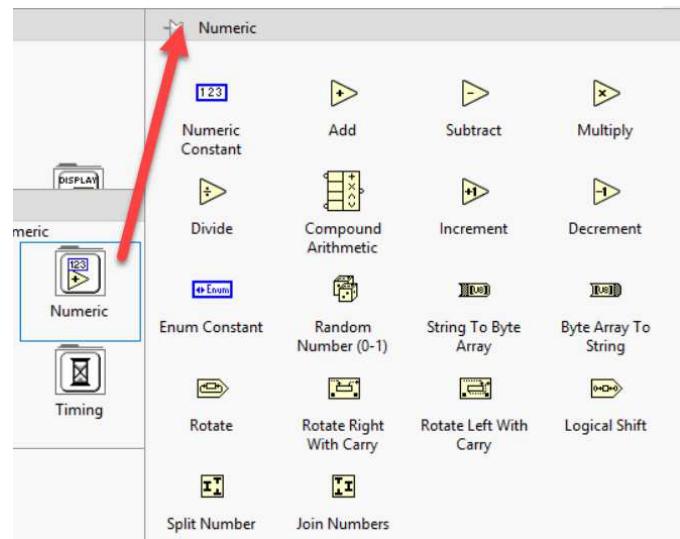


เป็น Reference ของ VI วันนี้ โดยจะเรียกใช้ในโปรแกรมหลักได้อย่างสะดวก เป็นส่วนของการซ้ำๆ หรือ นำไปที่ VI คล้ายกับการเรียกใช้งาน Sub Function ที่มีไฟล์นามสกุล .vi ขึ้นมาเพื่อใช้งาน

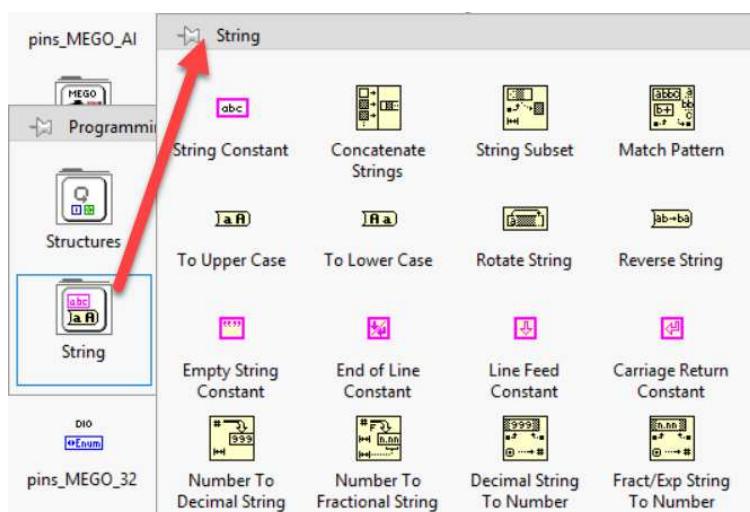
Array เป็นฟังก์ชันที่เกี่ยวกับการตัวแปรของข้อมูลชนิด Array (อะเรย์)



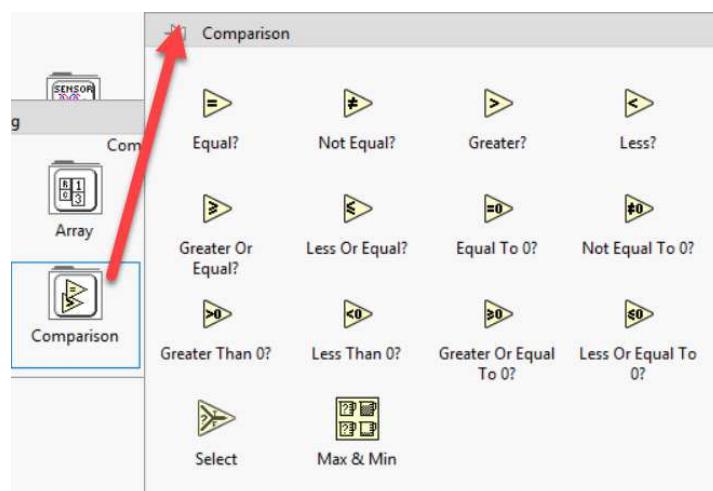
Numeric เป็นฟังก์ชันที่เกี่ยวกับการตัวแปรของข้อมูลชนิด Numeric (ตัวเลข)



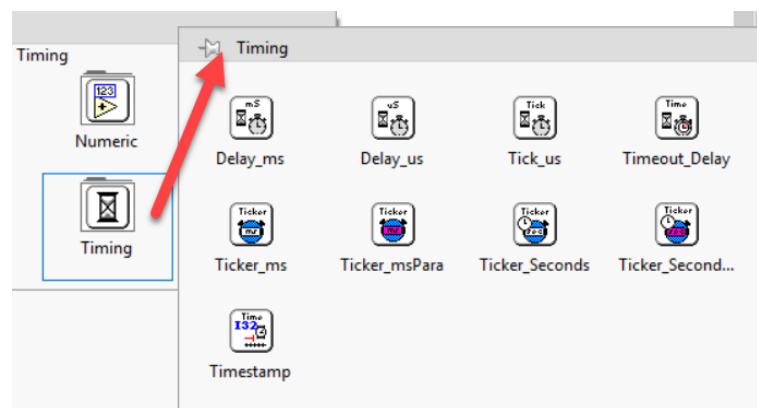
String เป็นฟังก์ชันที่เกี่ยวกับการตัวแปรของข้อมูลชนิดข้อความ (ตัวอักษร)



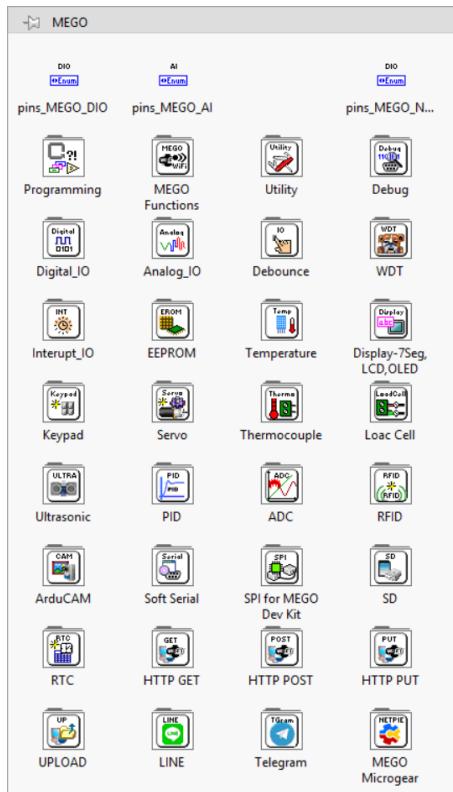
Comparison เป็นฟังก์ชันที่เกี่ยวกับการคำนวณ และการเปรียบเทียบ



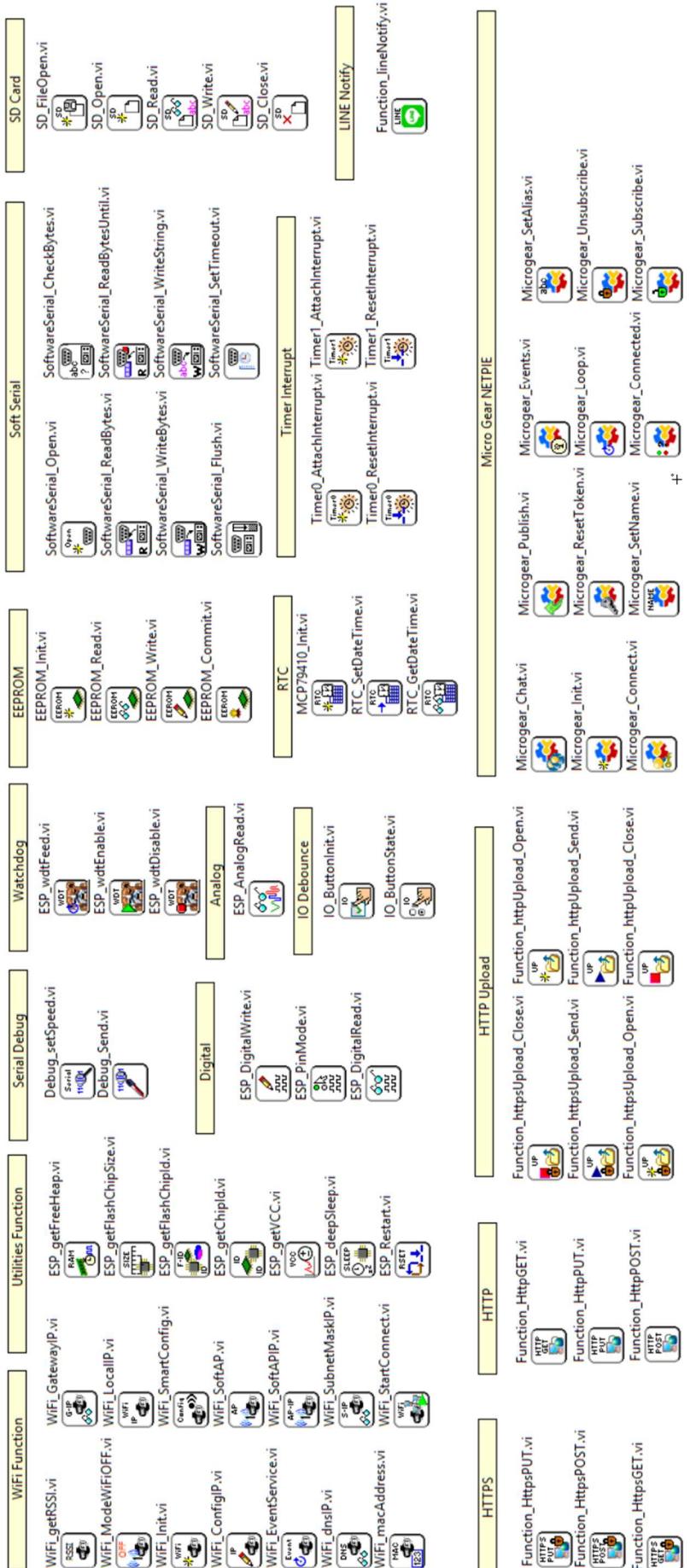
Timing เป็นฟังก์ชันที่เกี่ยวกับการจัดการด้านเวลา Timing

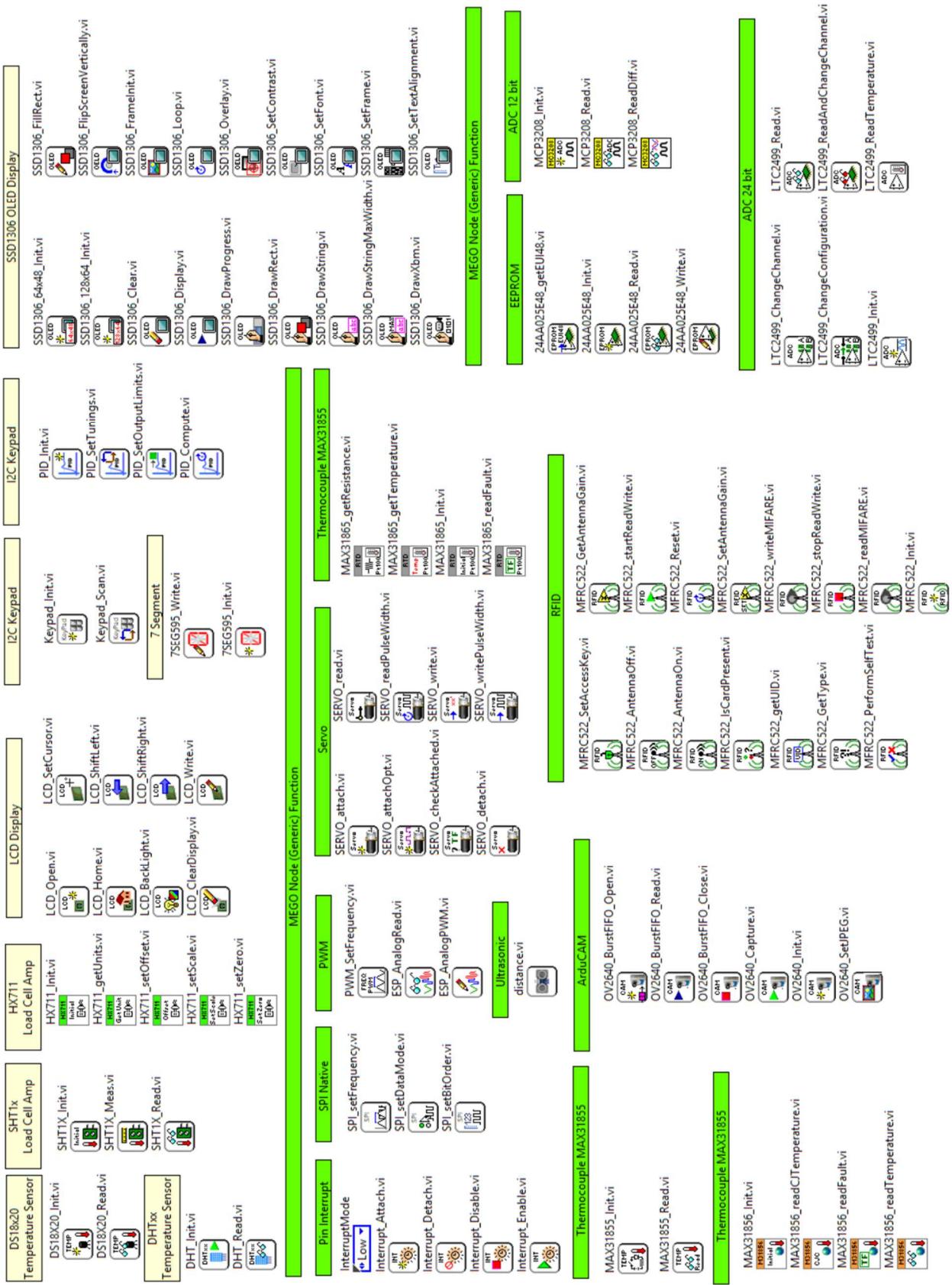


## MEGO Hardware Functions



ฟังก์ชันที่ติดต่อกับฮาร์ดแวร์ที่รองรับ สามารถเปลี่ยนหน่วยได้ดังนี้



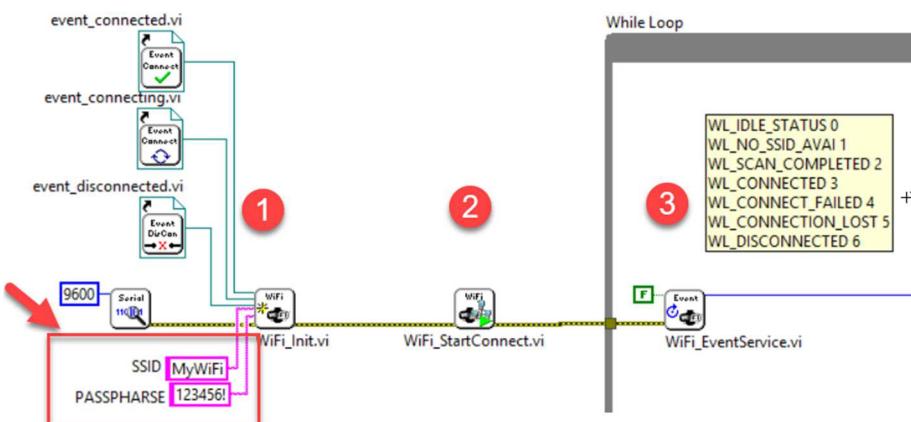


## WiFi Function



เป็นฟังก์ชันด้านการจัดการการเชื่อมต่อ WiFi และการอ่านค่าพารามิเตอร์ต่าง ๆ ของ WiFi โดยมีรูปแบบการใช้งานฟังก์ชันต่าง ๆ แบ่งได้ 3 รูปแบบ คือ

การเชื่อมต่อ WiFi แบบกำหนดค่า SSID และ Password



1. เรียกใช้ฟังก์ชัน WiFi\_Init.vi เพื่อกำหนดค่า SSID และ Password โดยฟังก์ชันนี้จำเป็นต้องเรียกใช้งานทุกครั้งก้าวต้องการใช้งาน WiFi ในการนี้ที่ไม่ต้องการใช้งานไปจำเป็นต้องเรียกขึ้นมาใช้ เพราะจะทำให้เก็บทรัพยากรด้าน RAM ของ MCU พoSmcw

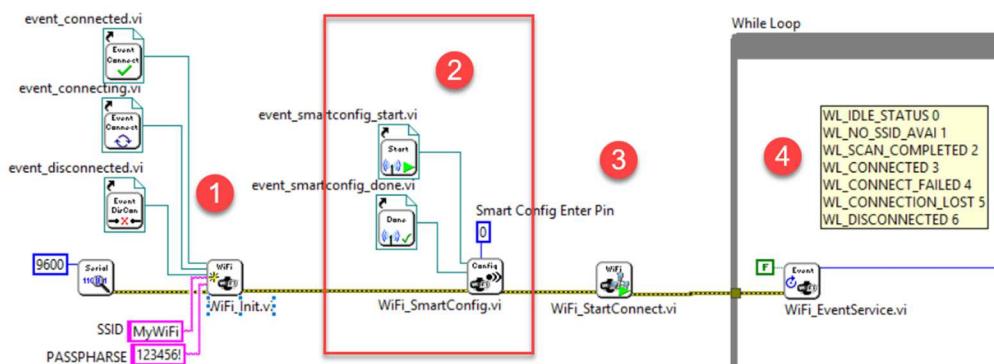
2. เรียกใช้ฟังก์ชัน WiFi\_StartConnect.vi เพื่อสั่งให้เชื่อมต่อ WiFi ตามที่กำหนด

3. เรียกใช้ฟังก์ชัน WiFi\_EventService.vi ใน While Loop โดยต้องเรียกเป็นฟังก์ชันแรกใน Loop โดยจะตรวจสอบสถานะการเชื่อมต่อ WiFi ตลอดเวลา ซึ่งสามารถตรวจสอบสถานะได้ และยังนำค่าที่ได้มา กำหนดเงื่อนไขการทำงานได้ โดยมีรายละเอียดดังนี้



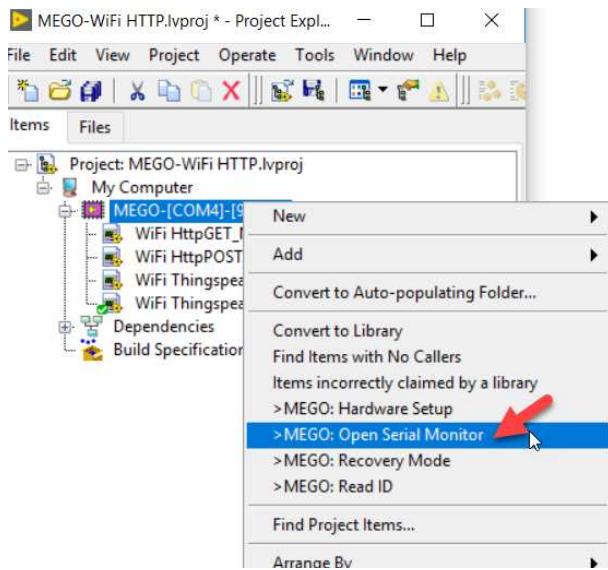
- 0 = WL\_IDLE\_STATUS – การเชื่อมต่อไม่สมบูรณ์
- 1= WL\_NO\_SSID\_AVAI – ไม่สามารถเชื่อมต่อกับ WiFi SSID อาจเกิดจากใส่ชื่อ SSID ผิด
- 2= WL\_SCAN\_COMPLETED - สถาบันค้นหา WiFi SSID เสร็จสมบูรณ์
- 3= WL\_CONNECTED – การเชื่อมต่อ WiFi สำเร็จ สามารถใช้งานได้กันที
- 4= WL\_CONNECT\_FAILED – การเชื่อมต่อ WiFi มีปัญหา อาจเกิดจากใส่ Password ผิด
- 5= WL\_CONNECTION\_LOST – การเชื่อมต่อผิดพลาด อาจจะเกิดจาก WiFi หายไปจากระบบ
- 6= WL\_DISCONNECTED - กำลังพยายามเชื่อมต่อ WiFi

การเชื่อมต่อ WiFi แบบ Smart Config โดยสามารถใช้ Smart Phone ป้อนค่า SSID และ Password

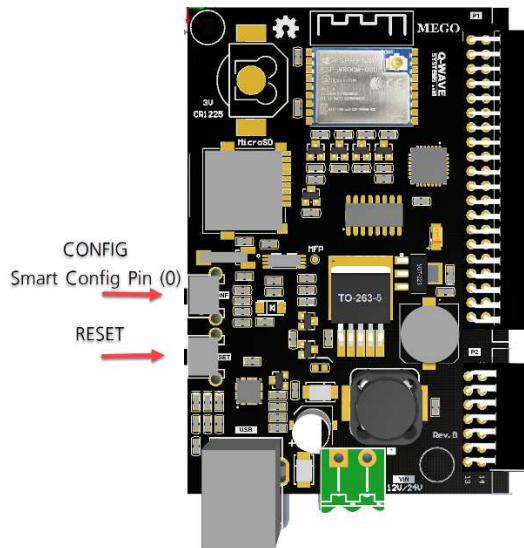


จะต้องเรียกใช้ฟังก์ชัน WiFi\_SmartConfig.vi ในขั้นตอนที่ 2. ตามรูป ซึ่งฟังก์ชันนี้จะเป็นการเปิดโหมดให้รองรับการตั้งค่า WiFi SSID และ Password ผ่านแอพพลิเคชันบนมือถือ โดยขั้นตอนการใช้งานดังนี้

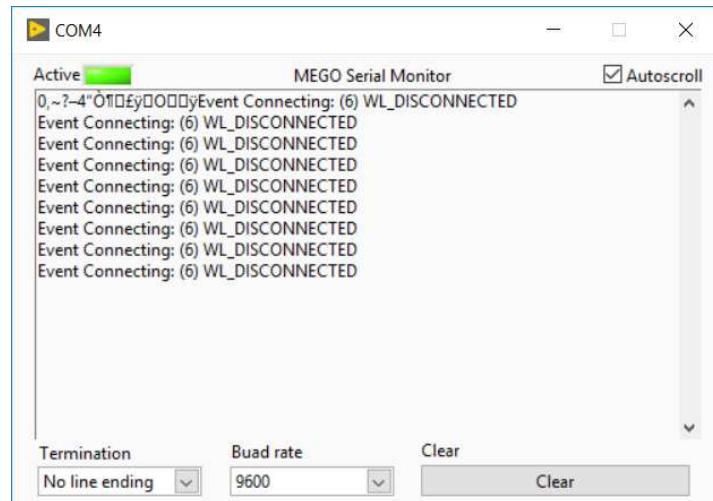
1. เปิด Serial Monitor เพื่อดูผลการทำงาน



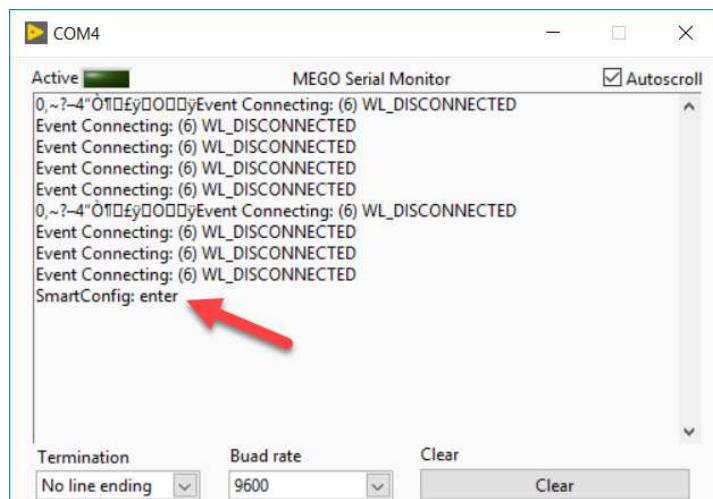
2.กดปุ่ม Reset ที่บอร์ด MEGO



3.ภายในเวลา 10 วินาที ในขณะที่แสดงผล “6= WL\_DISCONNECTED” บนหน้าต่าง Serial Monitor กดปุ่ม CONFIG บนบอร์ด MEGO



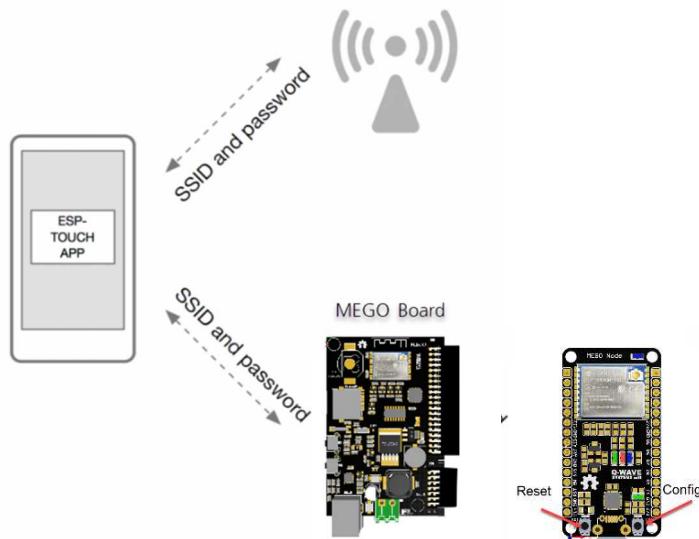
4. ถ้าผลการกำ้งานบน Serial Monitor จะแสดงคำว่า “Smart Config Enter” แสดงว่าบอร์ดพร้อมรอรับการตั้งค่าจาก Smart Phone



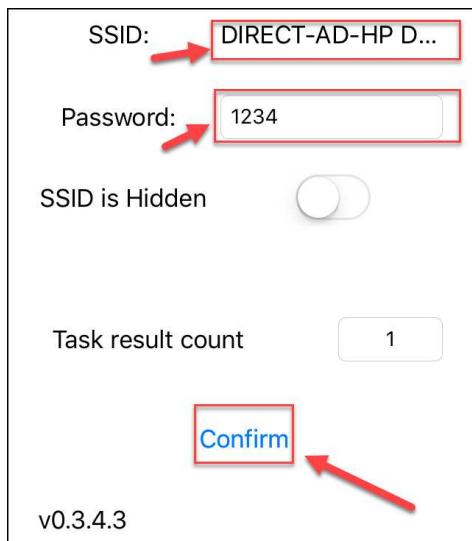
5. เปิดแอพพลิเคชัน ESP Smart Config บนมือถือ



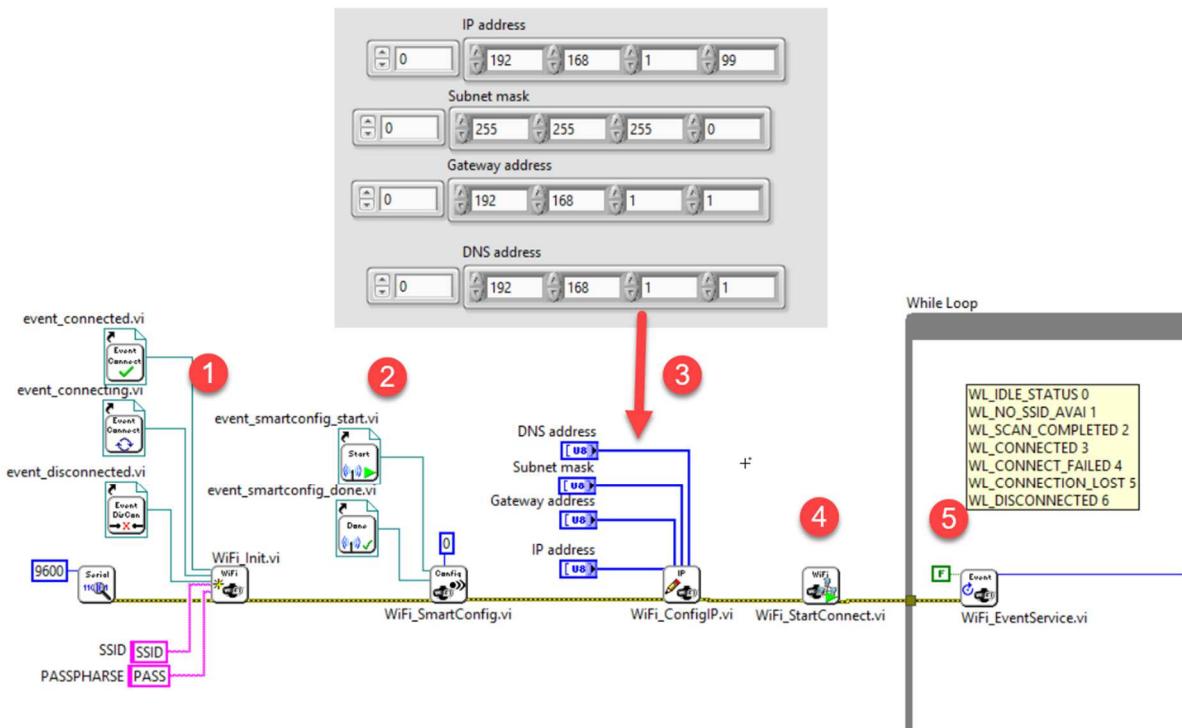
สำหรับ iOS ให้ดาวน์โหลด App ชื่อว่า “Esptouch”  
สำหรับ Andriod ให้ดาวน์โหลด App ชื่อว่า “ESP Smart Config”



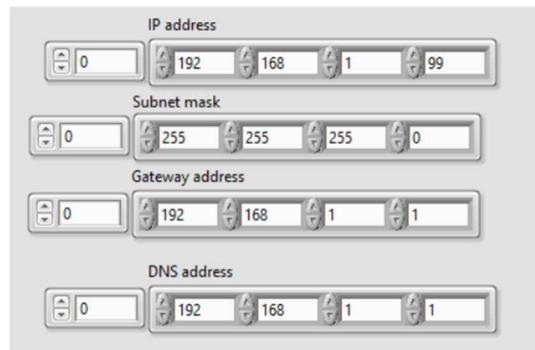
กด Confirm เพื่อกำกการ Config ในขั้นตอนนี้ WiFi SSID และ Password จะป้อนให้กับบอร์ด MEGO ผ่านทาง Smart Phone



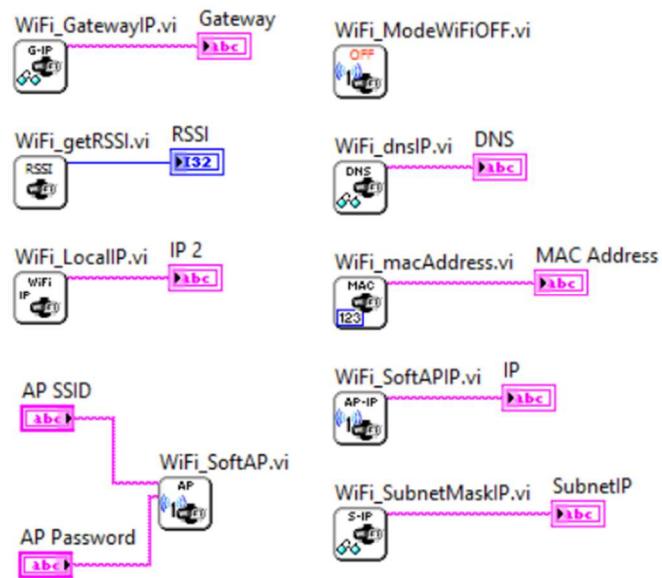
การเชื่อมต่อ WiFi โดยกำหนดค่า IP 固定 (Fixed IP)



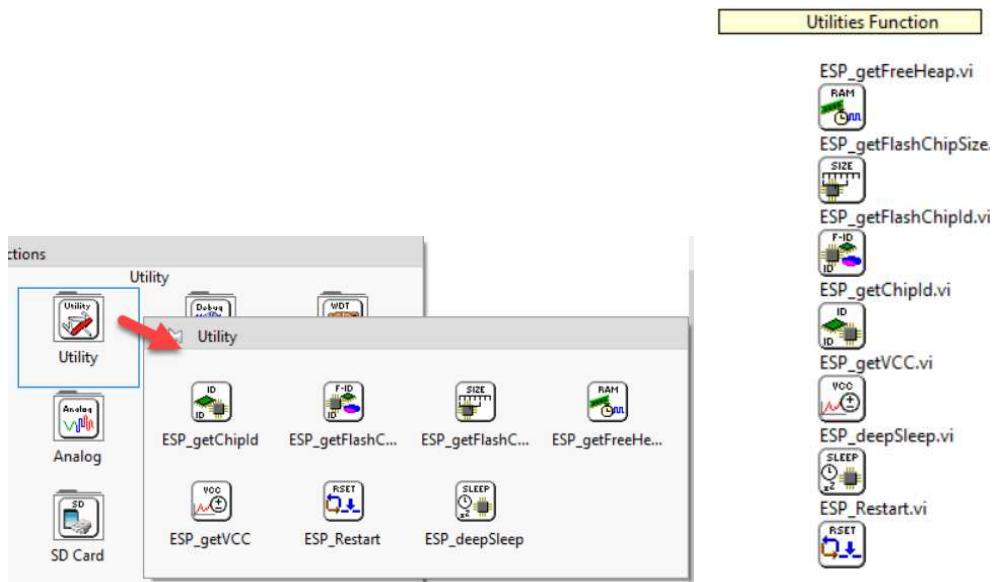
จะต้องเรียกใช้ฟังก์ชัน WiFi\_SmartConfig.vi ในขั้นตอนที่ 3. โดยสามารถกำหนด IP Address ได้เอง (Fixed IP)



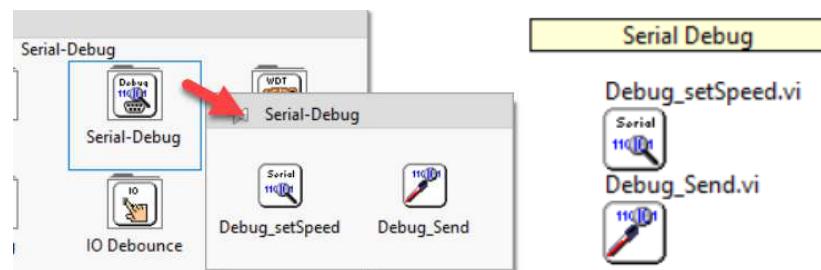
การใช้งานฟังก์ชันอ่านค่าพารามิเตอร์ WiFi อีน ๆ แสดงได้ดังนี้



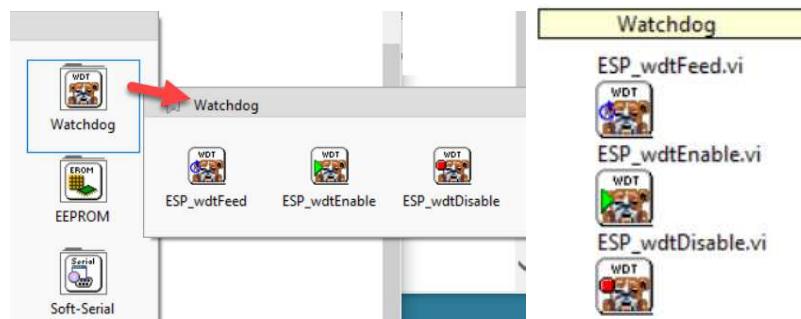
## Utility



## Serial Debug



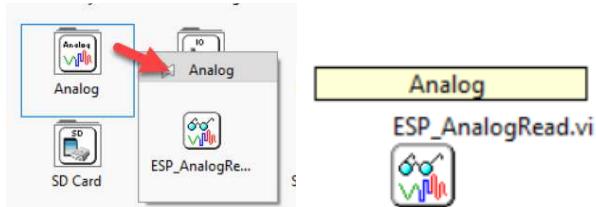
## Watchdog



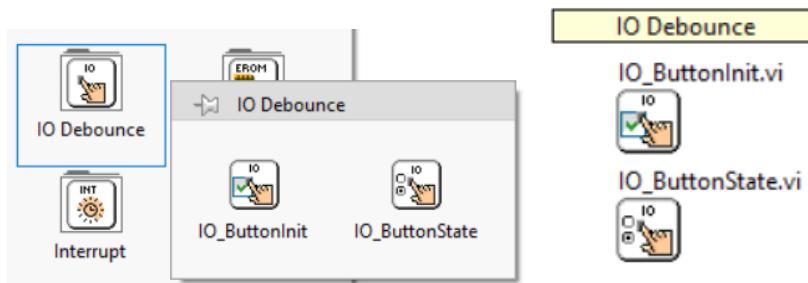
Digital



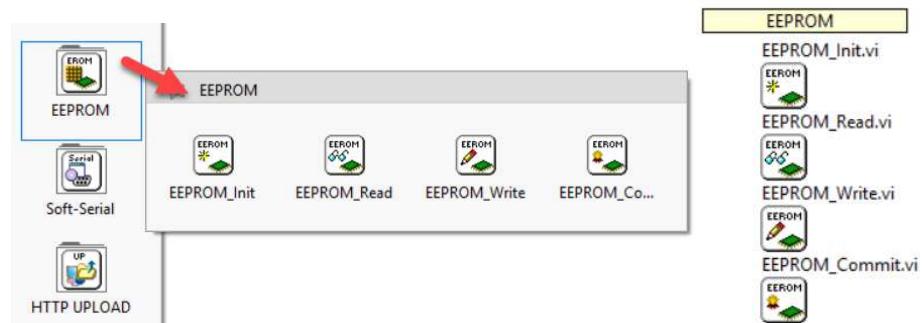
Analog



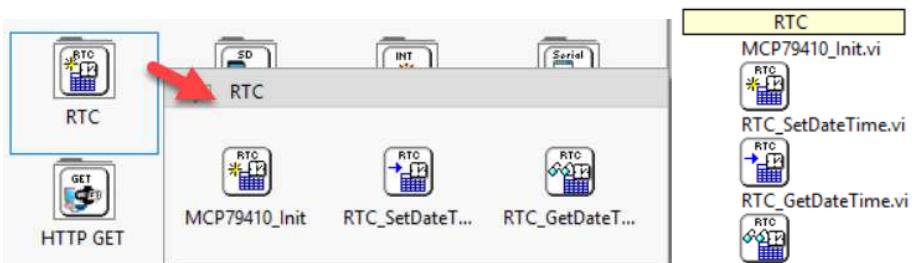
IO Debounce



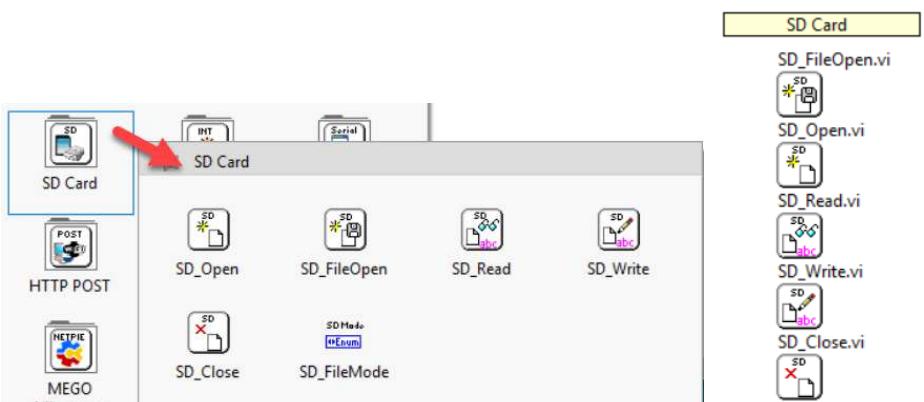
EEPROM



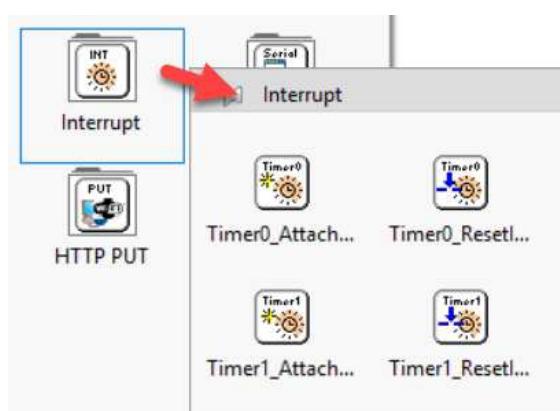
RTC

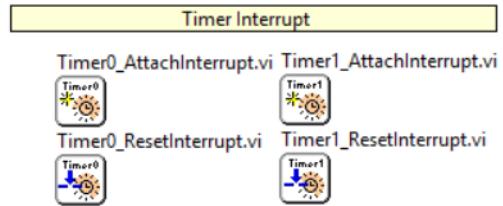


SD Card

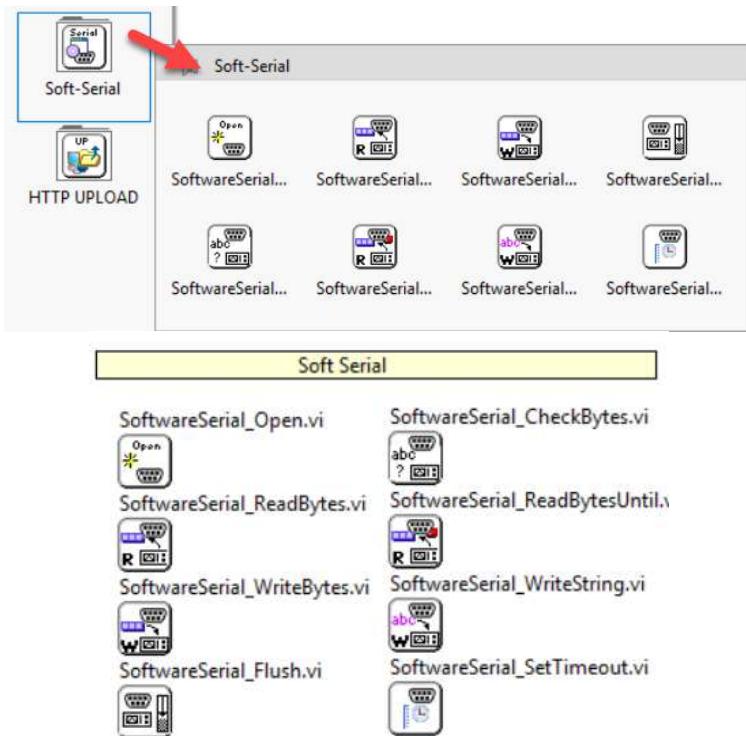


Interrupt

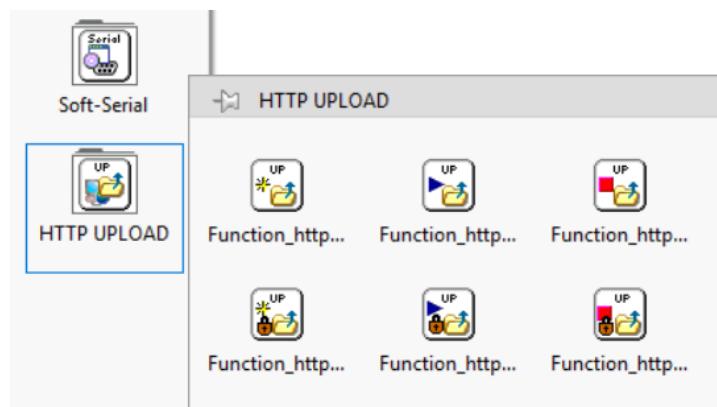


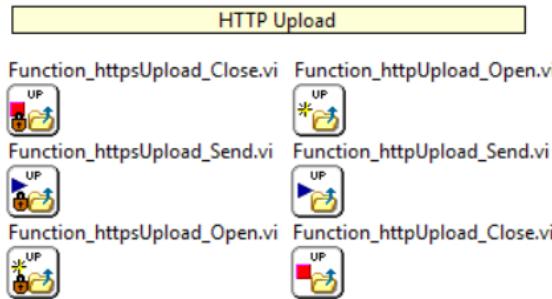


## Soft Serial

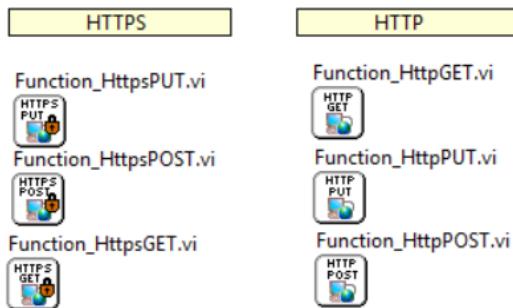
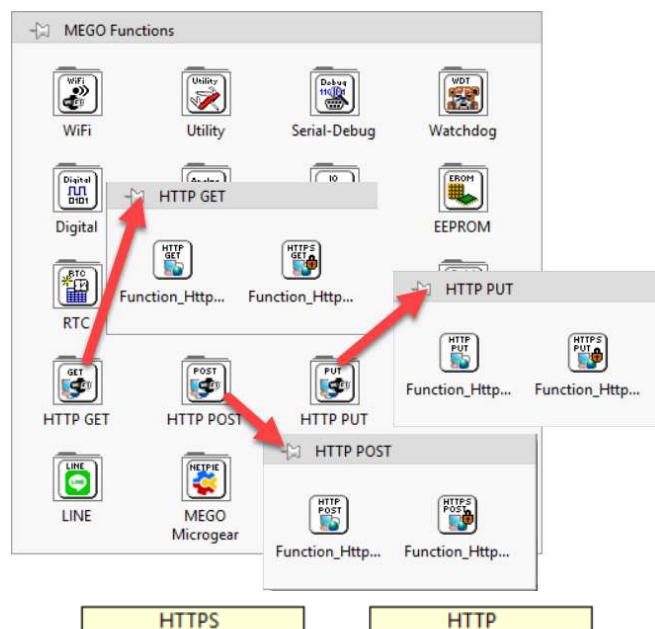


## HTTP Upload

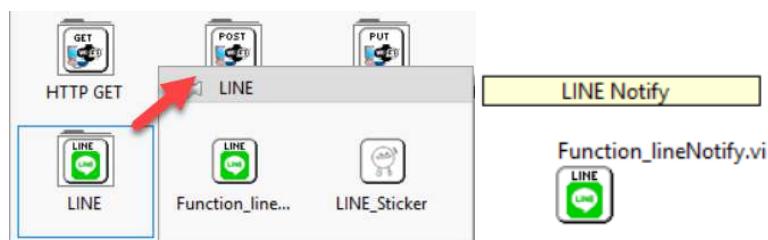




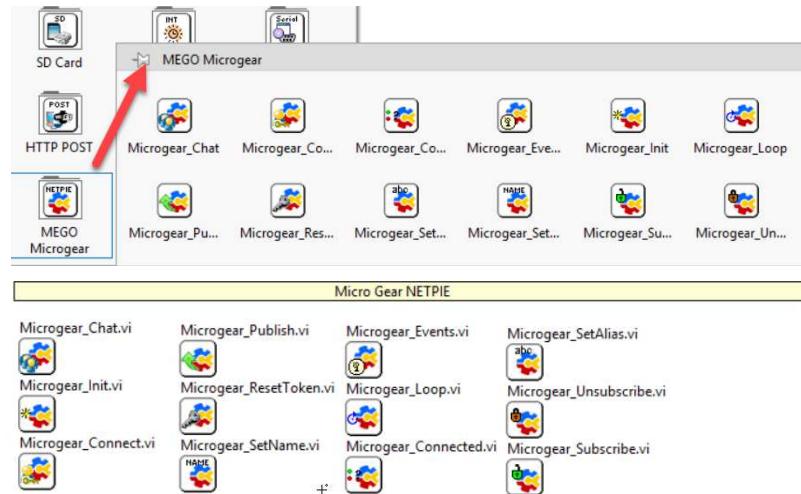
## HTTP/HTTPS



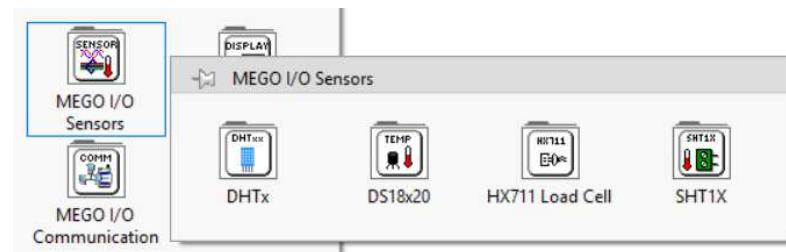
## LINE Notify API



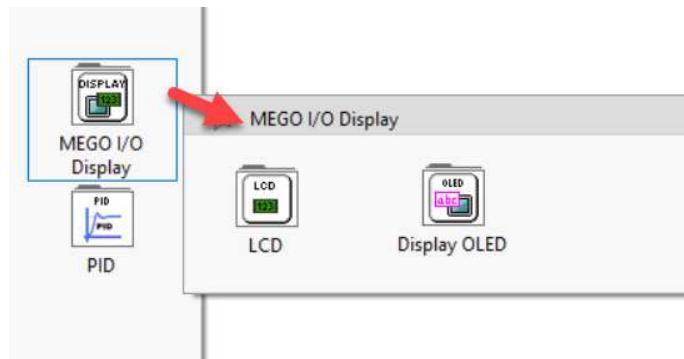
## Micro Gear (NETPIE)



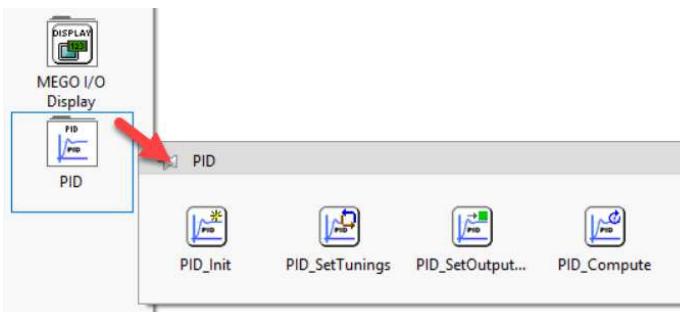
## MEGO Sensors



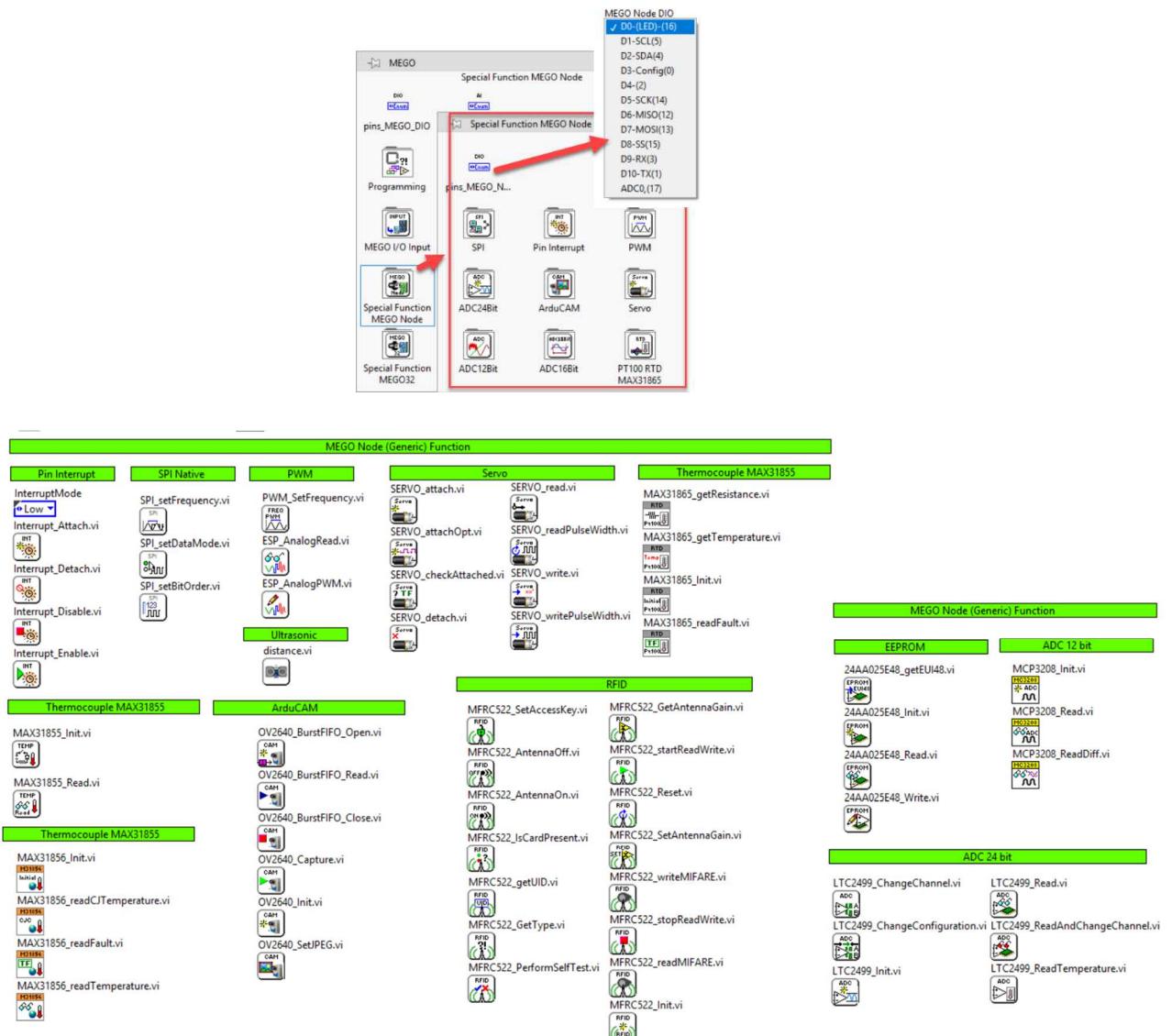
## MEGO Display



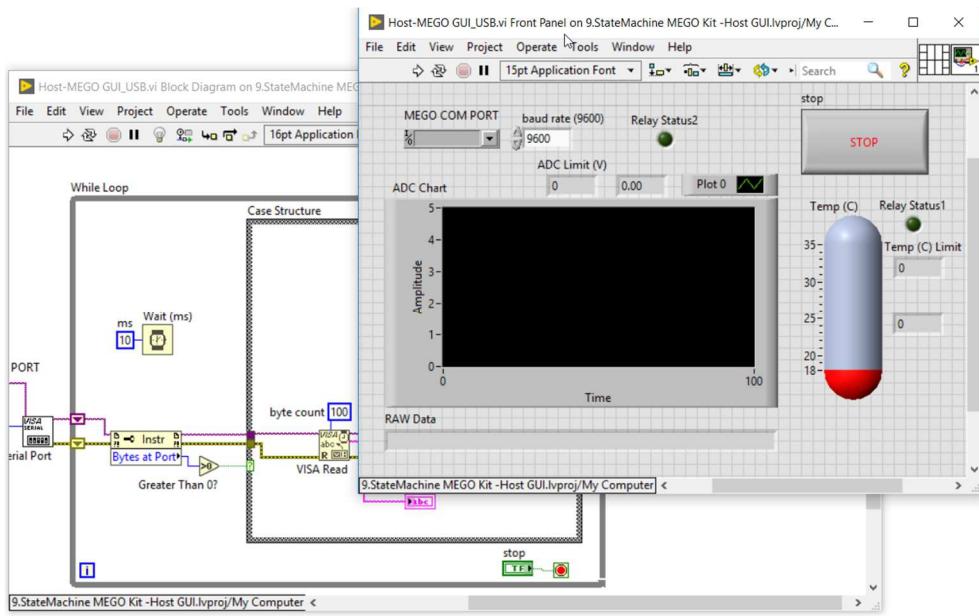
## PID Control



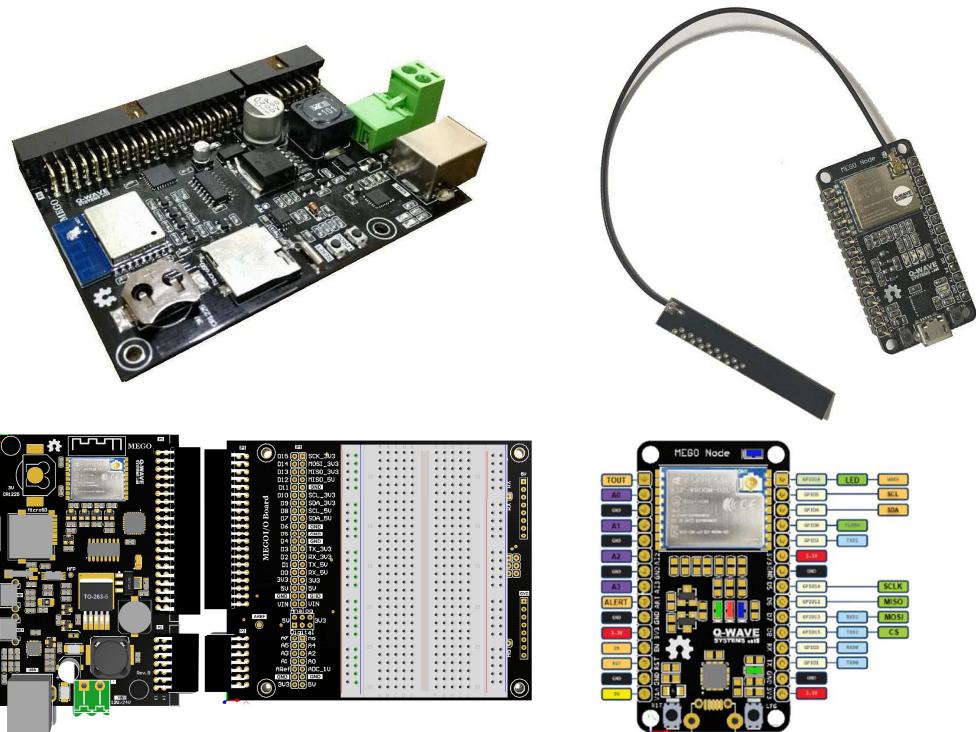
## Function ที่รองรับบอร์ด MEGO Node ได้บ้าง (\*เนื่องจากใช้ SPI Bus ในการสื่อสาร)



## การเขียนโปรแกรม LabVIEW เป็องตับ

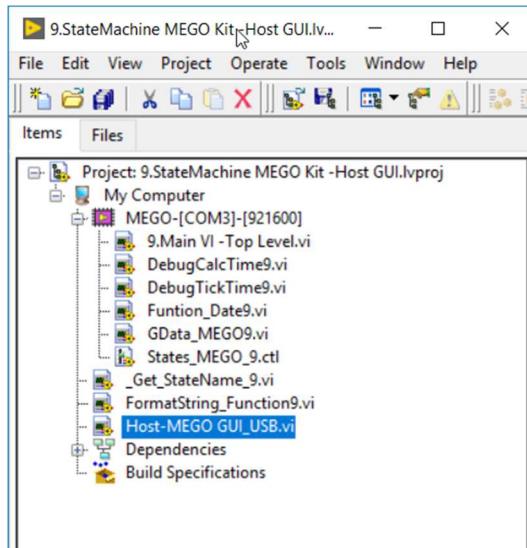


LabVIEW เป็นภาษาโปรแกรมในรูปแบบกราฟิก โดยสามารถเขียนโปรแกรมด้วยการลากวาง บล็อกฟังก์ชันต่าง ๆ ประกอบกันเป็นแอพพลิเคชัน และสามารถดาวน์โหลดโปรแกรมลงในเครื่องคอมพิวเตอร์ (MEGO) เพื่อทำงานแบบ Standalone หรือระบบสมองกลผังตัวได้



บอร์ด MEGO Dev Kit และ MEGO Node ที่สามารถโปรแกรมด้วย LabVIEW ได้

โดยจะมี 2 หน้างต่าง คือ 1.หน้าต่าง Block Diagram (สีขาว) และ 2.หน้าต่าง Front Panel (สีเทา) แสดงดังรูป

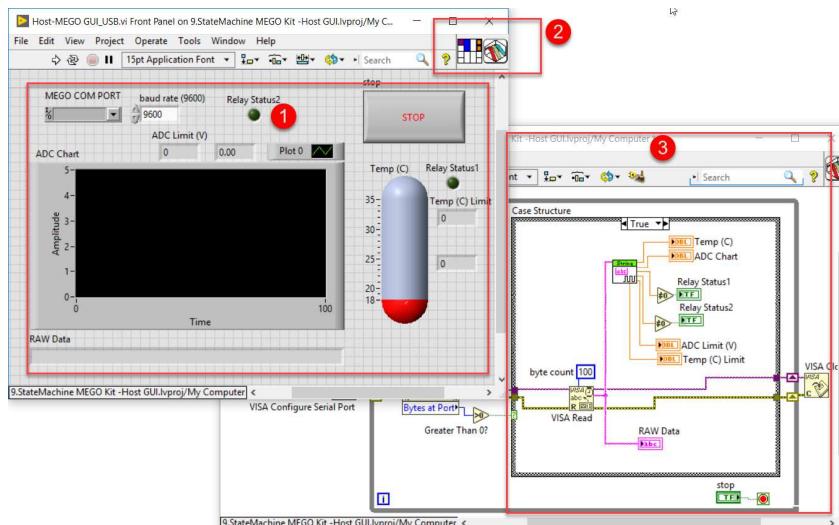


Project Explorer เป็นหน้าต่างจัดการไฟล์ต่างๆ ใน LabVIEW อาทิ ไฟล์ .vi หรือ .ctl สามารถสร้าง Folder เพื่อจัดเก็บไฟล์อย่างเป็นระบบ ทำหน้าที่ติดต่อ กับฮาร์ดแวร์ อาทิ การ Compile และ Download โปรแกรมลงบอร์ด MEGO ไฟล์ชนิดต่างๆ ที่สำคัญในโปรแกรม LabVIEW มีดังนี้

LabVIEW Project .lvproj คือ ไฟล์ที่เก็บคุณลักษณะของไฟล์ต่างๆ และฮาร์ดแวร์ที่ใช้งาน

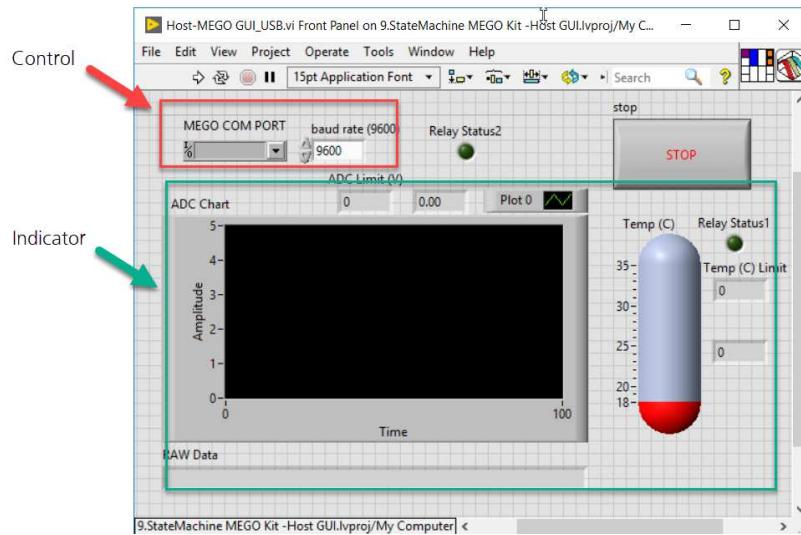
Virtual instrument (VI) .vi คือ โปรแกรม LabVIEW

Custom control .ctl คือ ไฟล์ที่เก็บคุณลักษณะ User Interface ที่สร้างขึ้นมา

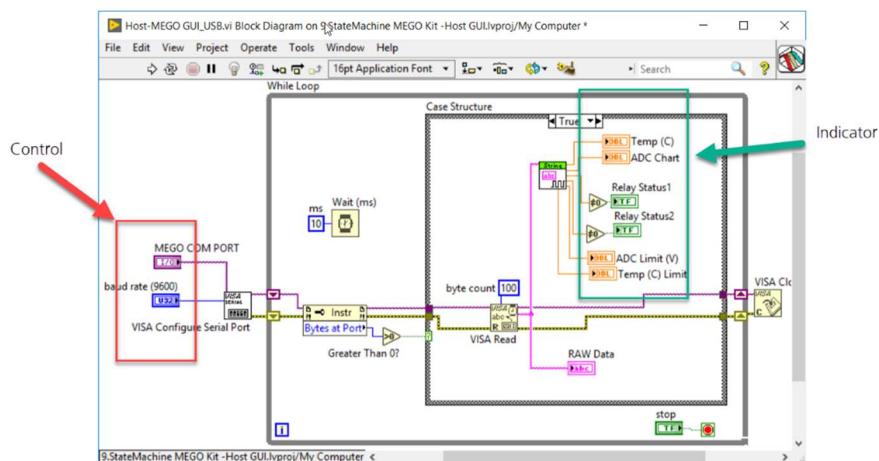


ส่วนประกอบของไฟล์ VI ประกอบไปด้วย 3 ส่วนคือ

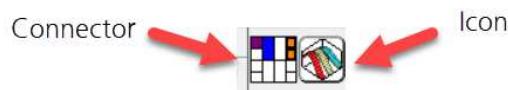
- 1.หน้าต่าง Front Panel (ส่วนติดต่อผู้ใช้งาน User Interface)
- 2.ส่วนของ Icon และ Connector (ส่วนเชื่อมต่อของ VI ในการที่นำไปใช้เป็นโปรแกรมย่อย SubVI)
- 3.หน้าต่าง Block Diagram (ส่วนโปรแกรม Source Code)



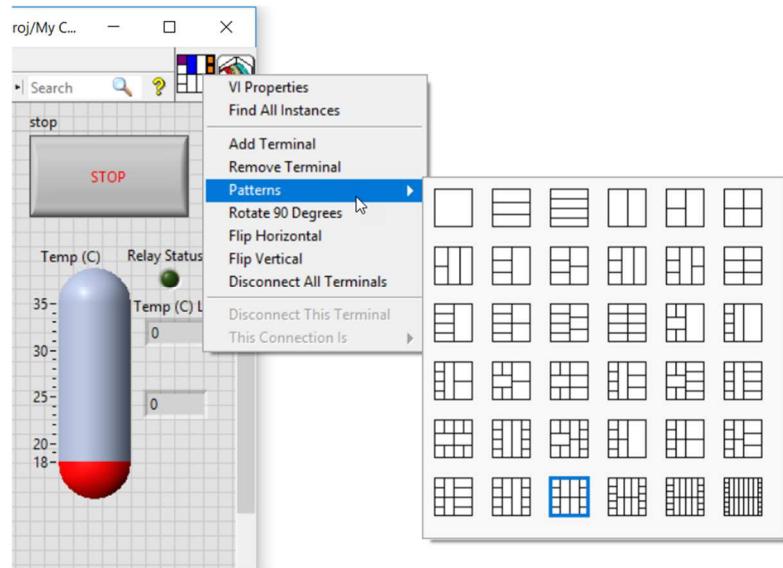
หน้าต่าง Front Panel เป็นส่วนติดต่อผู้ใช้งาน User Interface ประกอบไปด้วย การรับค่า Input (เรียกว่า Control) และการแสดงผล (เรียกว่า Indicator)



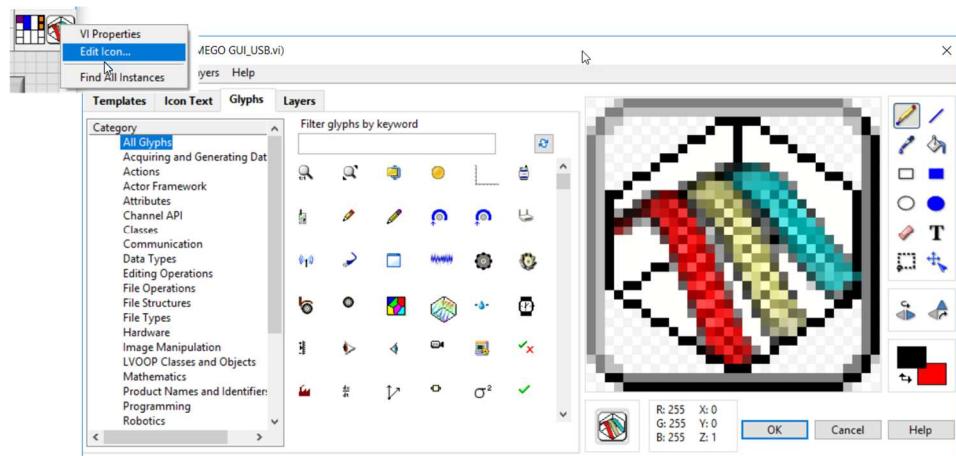
หน้าต่าง Block Diagram เป็นส่วนที่เขียนโปรแกรม Source Code ประกอบไปด้วยฟังก์ชันต่างๆ จากรูปจะเห็นว่าจะมีจุดเชื่อมต่อรับค่าอินพุต (Control) มาจาก Front Panel และในการส่งค่าเอ้าก์พุต ออกไปแสดงผล (Indicator) ที่ Front Panel โดยจุดเชื่อมต่อ Control และ Indicator ต่างๆ ใน Block Diagram จะเชื่อมโยงกับ Front Panel เสมือนว่าเป็นจุดเดียวกัน



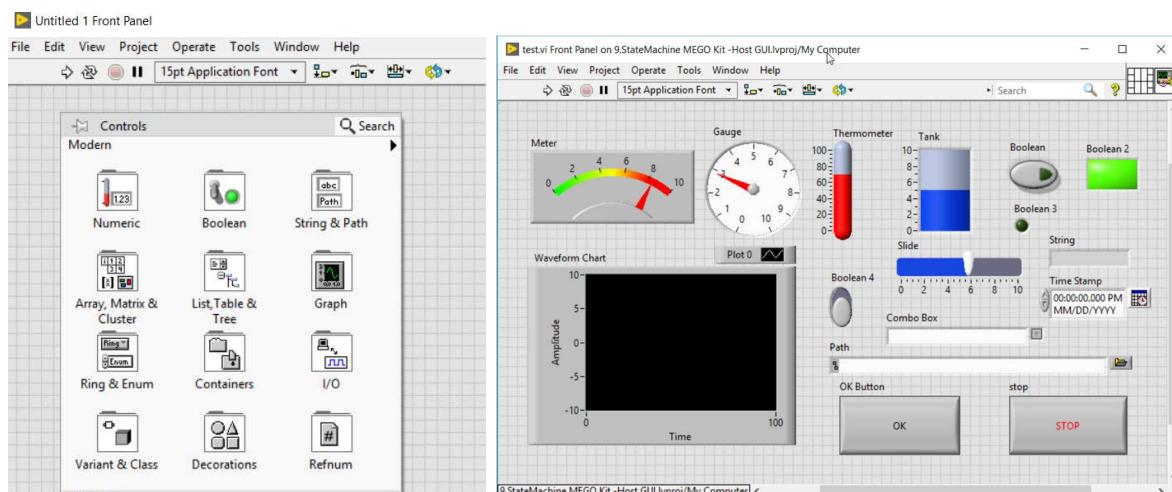
ไฟล์ .vi ทุกไฟล์จะต้องมีส่วนของ Icon/Connector โดยอยู่ที่นุ่มบนขวา ໃนส่วนของ Icon นั้นจะ กำหนดรูปรีบต้นมาให้แล้ว ไม่จำเป็นต้องสร้างขึ้นมาใหม่ แต่ในการออกแบบโปรแกรมที่ดี รูป Icon จะต้อง สื่อความหมายของโปรแกรมนั้นๆ เสมอ สำหรับ Connector มีความจำเป็นต้องกำหนดลักษณะเพื่อใช้งาน ในกรณีที่เราออกแบบเป็นโปรแกรมย่อย (SubVI) ที่อยู่ในโปรแกรมหลักไฟล์อื่น เสมือนว่าเราออกแบบ Sub Function แล้วเรียกใช้งานในโปรแกรมหลัก (Main VI)



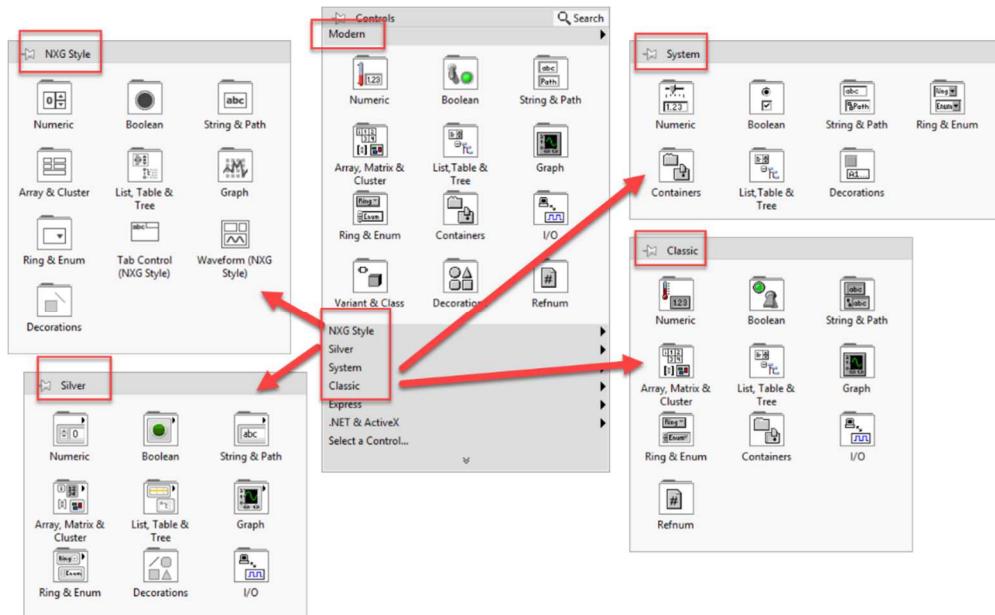
โดย Connector สามารถกำหนดลักษณะ (Pattern) ต่างๆ ได้หลายรูปแบบขึ้นอยู่กับจำนวนที่ต้องการ โดยให้คลิกขวาที่ Icon จากนั้นเลือก Pattern ที่ต้องการและดึงรูป



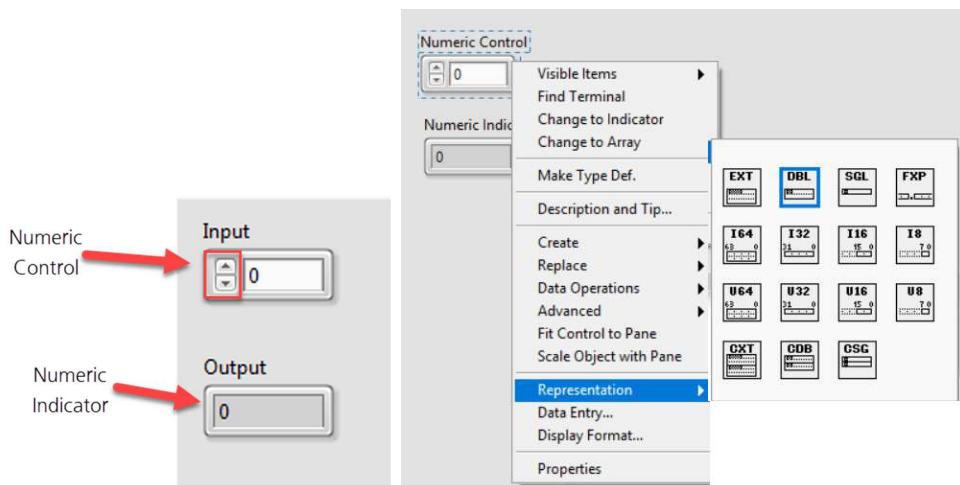
คลิกขวาที่รูป Icon มุมบนขวาของ VI เลือก Edit Icon จะเป็นการเปิดหน้าต่าง Icon Editor โดยเราสามารถสร้าง Icon รูปต่างๆ ได้ตามต้องการ จะเห็นว่ามีรูปต่างๆ พร้อมใช้งานกันที



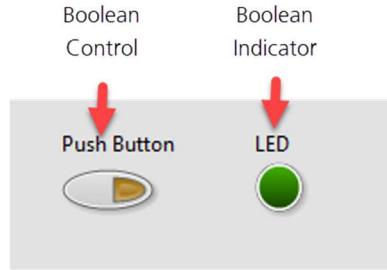
ໃນหน้าต่าง Front Panel ເຮັດວຽກຮ້າງ Control ແລະ Indicator ລາຄຫລາຍຮູປແບບຂັ້ນວູ່ແບບ  
ຂົດຂອງຂ່າມຸລ (Data Type) ໃນຮູບ ເປັນຕົວຢ່າງ ສ່ວນແສດງຜລ (Indicator) ທີ່ນີຍໃຊ້ງານໃນ LabVIEW



ຮູປແບບຂອງ Front Panel ມີໄ້ເລືອກຫລາຍແບບ ວັດທະນາ (Modern, NXG, Sliver, Systems) ແລະ Classic  
ໂດຍຄ່າເຮັດວຽກຈະກຳຫົວດັກຂະນະ Modern ມາໃໝ່ໃຊ້ງານແສດງດັ່ງຮູບ



ຕົວແປ່ນັດຕົວເລຂ (Numeric) ໃນหน้าຕ່າງ Front Panel ສາມາດເລືອກໃຊ້ງານໃໝ່ເປັນ Control ກັບ  
Indicator ກີ່ໄດ້ ແສດງດັ່ງຮູບ ໂດຍຕ້ອງກຳຫົວດັກຂົດຂອງຕົວເລຂໄດ້ ວັດທະນາ (ຈຳນວນເຕີມ) ກັບ floating-point (ຕົວເລຂມີຈຸດຖານຸຍິນ) ເພື່ອໃຊ້ງານ

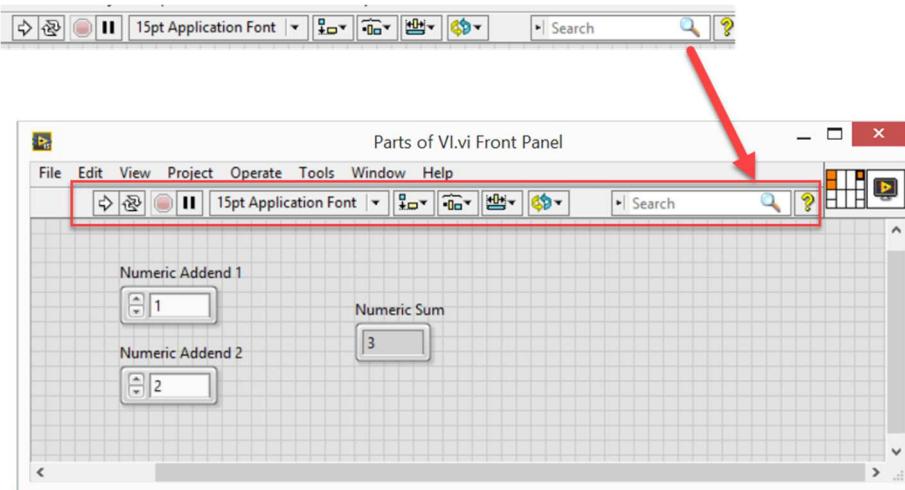


ตัวแปรชนิดบูลีน (Boolean) จะมี 2 สถานะ อาทิ True/False (จริง/เท็จ), 0/1 หรือ ON/OFF ถ้าต้องการติดต่อกับฮาร์ดแวร์เวย์ จะนิยมใช้ร่วมกับ Digital I/O เช่นการสั่ง ON/OFF เป็นต้น



ตัวแปรชนิด String ประกอบไปด้วยตัวอักษร (ASCII Characters) หลายๆตัวรวมกัน สามารถใช้งานแบบ Control (Input) หรือ Indicator (Output) ก็ได้ อาทิ ใช้งานเพื่อการรับค่า ข้อความจาก User Input หรือ รับค่า Password (Text) ในกรณีที่ใช้งานเป็น Indicator (Output) อาจจะใช้ในการแสดงผลข้อความต่างๆ ข้อความเดียว หรือข้อความแบบตาราง (Table) ก็ได้

เครื่องมือต่างๆ ในหน้าต่าง Front Panel และ Block Diagram



แกบเครื่องมือ (Toolbar) ในหน้าต่าง Front Panel เครื่องมือสำหรับรันโปรแกรม และการดีบักคิด มีรายละเอียดดังนี้

 ปุ่มรัน (Run) ใช้เพื่อสั่งให้โปรแกรมทำงาน กรณีที่ต้องการหยุด ผู้เขียนต้องสร้างปุ่ม Stop ขึ้นมาเอง (สำหรับโปรแกรมที่รันบน Windows เก่า�ัน แต่การเขียนโปรแกรมร่วมกับบอร์ด MEGO จะใช้วิธี Compile และ Download แทน)

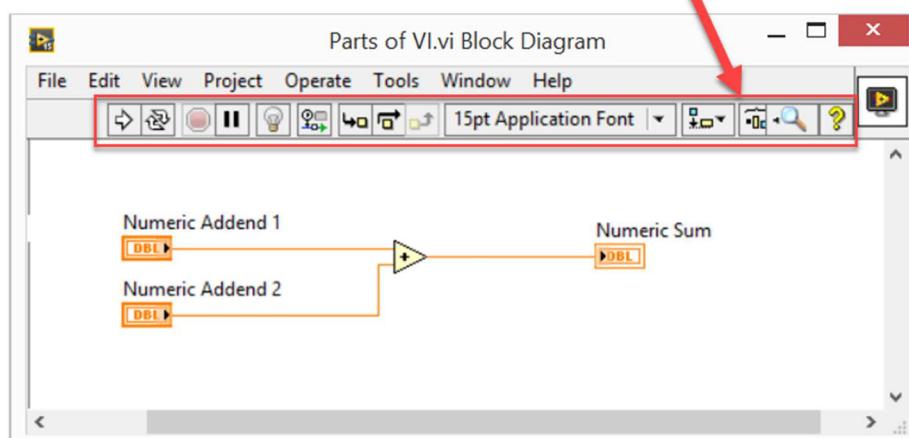
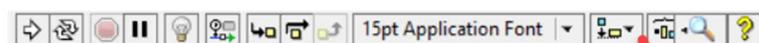
 ปุ่มรันแบบต่อเนื่อง (Run Continuously) ใช้เพื่อสั่งให้โปรแกรมทำงานแบบต่อเนื่อง ถ้า

ต้องการหยุด  ให้กดปุ่มหยุดโปรแกรม Abort Execution โดยมีym ใช้เพื่อกดสอบโปรแกรม เก่า�ัน ไม่มีym ทำไปใช้งานจริง

 ปุ่มหยุดโปรแกรมชั่วคราว Pause ใช้เพื่อยุดโปรแกรมในขณะที่ทำงาน เพื่อจะ Debug โปรแกรม

 เครื่องมือแก้ไขรูแบบตัวอักษร ขนาด และสีของข้อความ  
 เครื่องมือจัดเรียงวัตถุต่างๆ ใน Front Panel อาทิ จัดเรียงแนวตั้ง แนวนอนเป็นต้น

 เครื่องมือค้นหา Search โดยสามารถค้นหาฟังก์ชัน ตัวอย่าง โปรแกรม หรือข้อมูลนั้นทางออนไลน์ได้ด้วย



ในหน้าต่าง Block Diagram จะมีแถบเครื่องมือ (Toolbar) คล้ายกับบน Front Panel มีรายละเอียด พอกันเช่นเดียวกัน

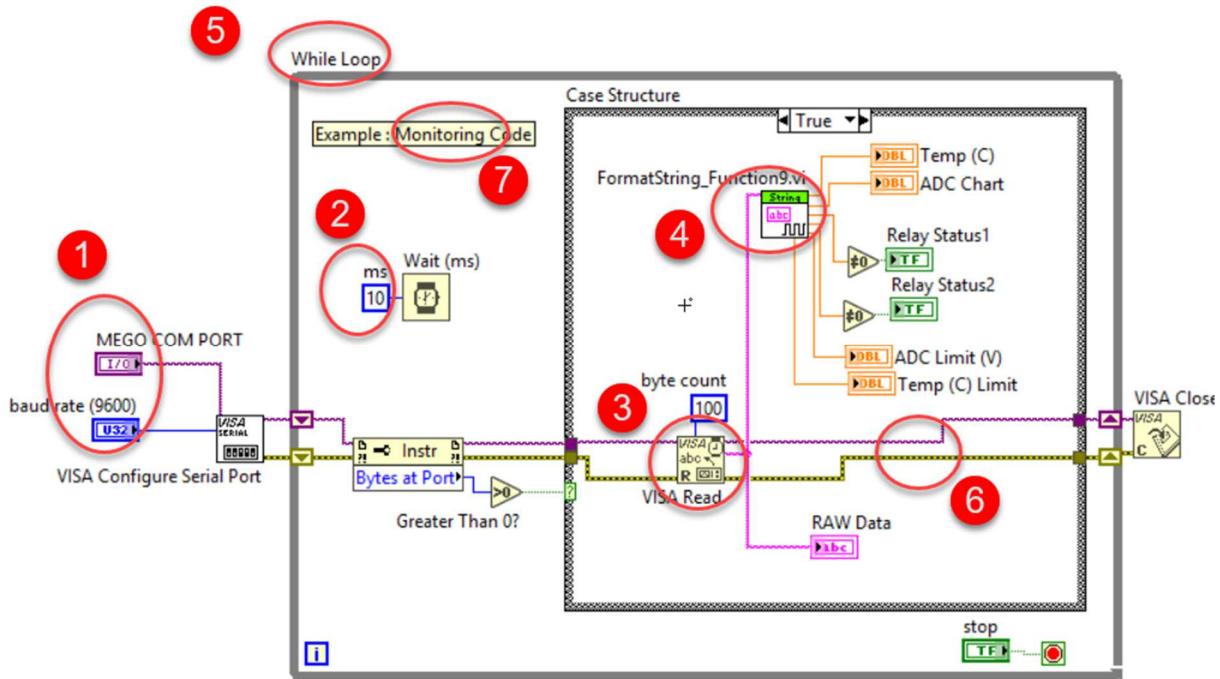
 Highlight Execution ช่วยในการ Debug โปรแกรม โดยเปิดการมองเห็นข้อมูลที่ว่างผ่านจุดต่างๆใน ขณะที่ทำงาน ซึ่งมีประโยชน์มากในการตรวจสอบการทำงานของโปรแกรม (Debug) ว่า ทำงานงานได้ตามที่ออกแบบหรือไม่

 Retain Wire Values สั่งให้บันทึกข้อมูลแต่ละจุดของโปรแกรม เพื่อสามารถดูข้อมูลย้อนหลังได้ เพื่อช่วยในการ Debug ข้อมูลได้ในภายหลัง การทำงานของโปรแกรมจะช้าลง เพราะว่า ต้องทำงานและบันทึกข้อมูลไปด้วย การเรียกคืนผลลัพธ์หลังจากได้โดยการเรียกฟังก์ชัน Probe ดูผลในจุดต่างๆของโปรแกรม



Step Into, Step over, Step Out เปิดการสั่งให้โปรแกรมแบบ Manual ที่ลະขันตอนซึ่งโดยผู้ใช้กำหนดเอง ซึ่งจะมีประโยชน์ในการ Debug โปรแกรมอย่างมาก พังก์ชั่นนี้สมควรกับเครื่องมือ Debug ในภาษาอื่น

ส่วนประกอบต่าง ๆ ของ Block Diagram

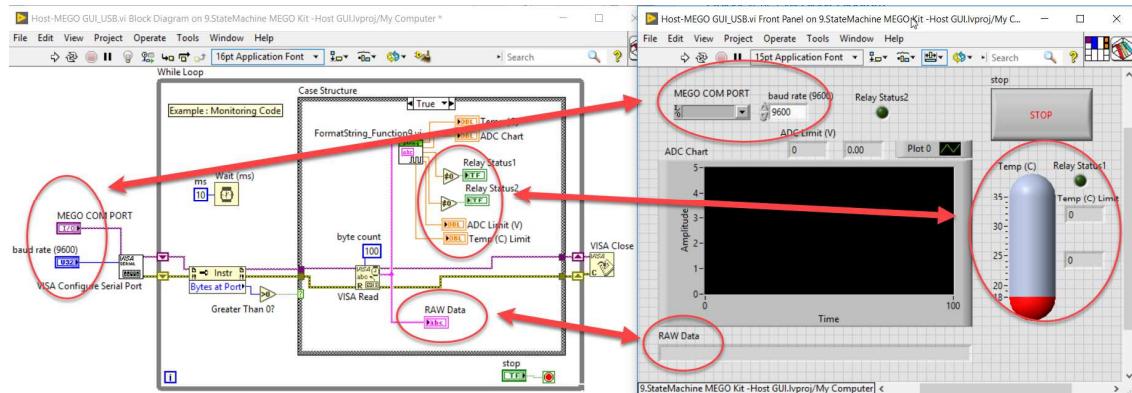


- ① Terminals: เป็นจุดเชื่อมต่อข้อมูลระหว่าง Block Diagram และ Front Panel โดยแต่ละ Terminal จะเป็น Control หรือ Indicator ได้
- ② Constant: เป็นตัวแปรคงที่ ที่ประกาศในโปรแกรม เป็นการสร้างและเรียกใช้งานใน Block Diagram เก่า�ัน ผู้ใช้งานไม่สามารถแก้ไขข้อมูลในหน้าต่าง Front Panel ได้ในขณะที่โปรแกรมกำ(SIG)
- ③ Function: เป็นฟังก์ชันพื้นฐานที่มีมาให้เลือกใช้ใน LabVIEW โดยสังเกตว่า ปกติฟังก์ชันจะมีสีเหลือง และไม่สามารถเปิดดู Block Diagram ของฟังก์ชันนั้นๆ ได้
- ④ SubVI: เป็นฟังก์ชันที่ผู้พัฒนาโปรแกรมสร้างขึ้นมาเอง มักเรียกใช้งานเป็นโปรแกรมย่อยในโปรแกรมหลัก สามารถออกแบบลักษณะ Icon ของฟังก์ชันได้เอง
- ⑤ Structure: เป็นฟังก์ชันพื้นฐานของ LabVIEW ที่ใช้งานในลักษณะ Structure อาทิ While Loop, For Loop, Flat Sequence Structure เป็นต้น
- ⑥ Wire: เป็นการส่งค่าระหว่าง Node ต่างๆ ใน LabVIEW โดยสืบของ Wire จะบ่งบอกชนิดของข้อมูล

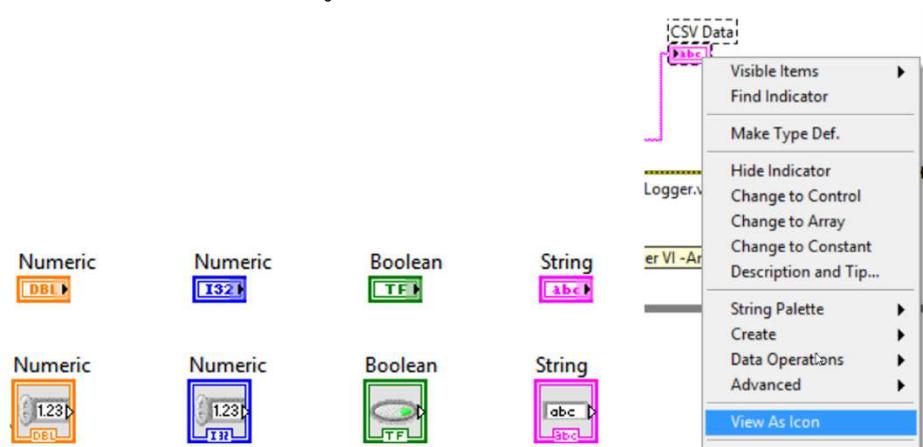
7

Free Labels: เป็นคำอธิบายโปรแกรม กีสร้างขึ้นเพื่ออธิบายความหมายของโปรแกรมในจุดต่างๆ ได้ เสมือนการเขียน Comment ในภาษา Text Program อีก

## Terminal

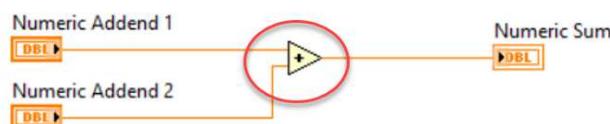


ตัวอย่าง Terminals ที่เป็นจุดเชื่อมต่อข้อมูลระหว่าง Block Diagram และ Front Panel เสมือนว่าเป็นจุดเดียวกันของโปรแกรม แสดงได้ดังรูป

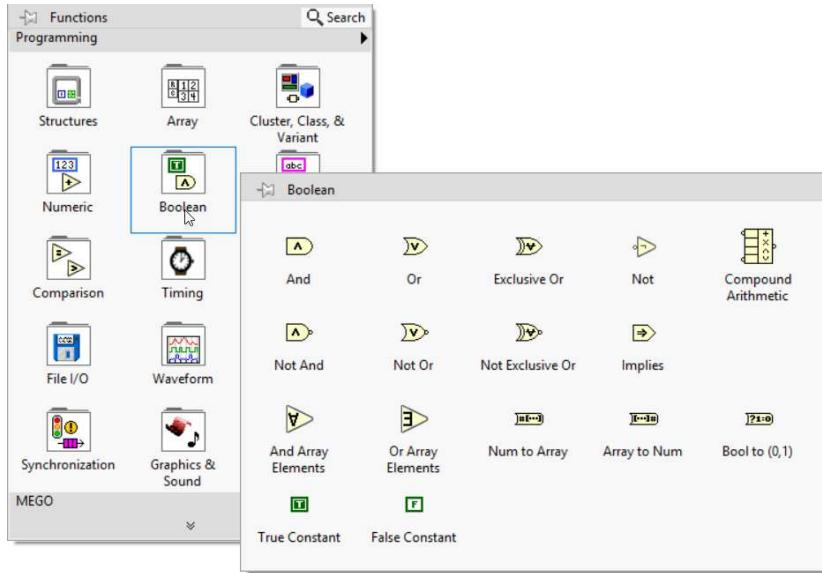


Terminal ต่างๆ ใน Block Diagram จะแสดงเป็นสีต่างๆ เพื่อบ่งบอกถึงชนิดของข้อมูล Data Type และสามารถกำหนดให้แสดงผลในลักษณะ Icon ได้ด้วย โดยคลิกขวาที่ Icon เลือก View As Icon

## Functions



ฟังก์ชันต่างๆ กีเรียกใช้งานในโปรแกรมเรียบง่ายเรียกว่า Node เสมือนว่าเป็นจุดเดียวของโปรแกรม จำกัดตัวอย่างเป็นการบวกตัวเลขสองตัว โดยฟังก์ชัน Add มีสีเหลืองดังรูป เรียกว่า Function Node เป็นฟังก์ชันพื้นฐานกีมีมาให้เรียกใช้ใน LabVIEW

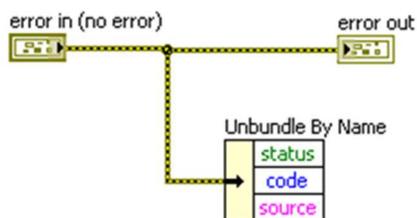


การเรียกใช้งาน Function Node ตัวอย่างการเรียกฟังก์ชัน Boolean เกี่ยวกับการจัดการกับข้อมูลชนิด Boolean ||| แสดงดังรูป

### Wire

	Boolean	Integer	String	Floating-point
Scalar	.....	—	~~~~~	—
1-D Array	.....	—	oooooooooo	—
2-D Array	.....	—	RRRRRRRRRR	—

Wire เป็นการส่งข้อมูลของโปรแกรม โดยจะส่งค่าจากจุดหนึ่งไปยังอีกจุดหนึ่ง เป็นเส้นอ่อน สายไฟ จุดเด่นของภาษา LabVIEW คือ Wire เพราะว่าเราสามารถ Debug จุดต่าง ๆ ของโปรแกรม โดย บนบีเตอร์ข้อมูลที่วิ่งผ่าน Wire ในโปรแกรม โดย Wire จะมีสี ขนาด และความหนาของเส้น เป็นตัวกำหนด Data Type ชนิดต่าง ๆ



สีเขียว คือข้อมูล (Data Type) ชนิดบูลีน Boolean

สีน้ำเงิน คือข้อมูล (Data Type) ชนิดตัวเลข จำนวนเต็ม (Integer)

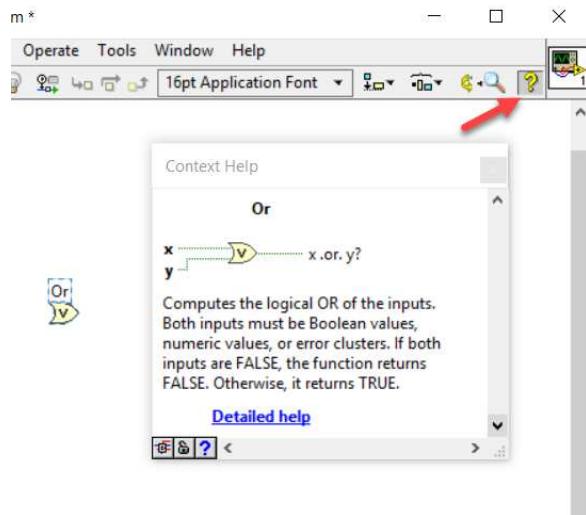
สีส้ม คือข้อมูล (Data Type) ชนิดตัวเลข จำนวนบทศิยม (Floating Point)

สีชมพู คือข้อมูล (Data Type) ชนิดตัวอักษร (String)

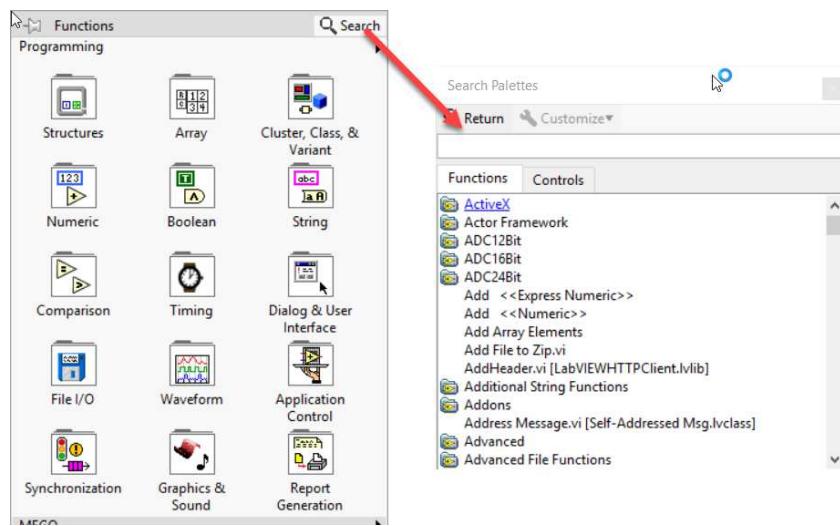
สีเหลือง คือข้อมูล (Data Type) ชนิดพิเศษเรียกว่า Error ประกอบไปด้วย Status, Code, Source



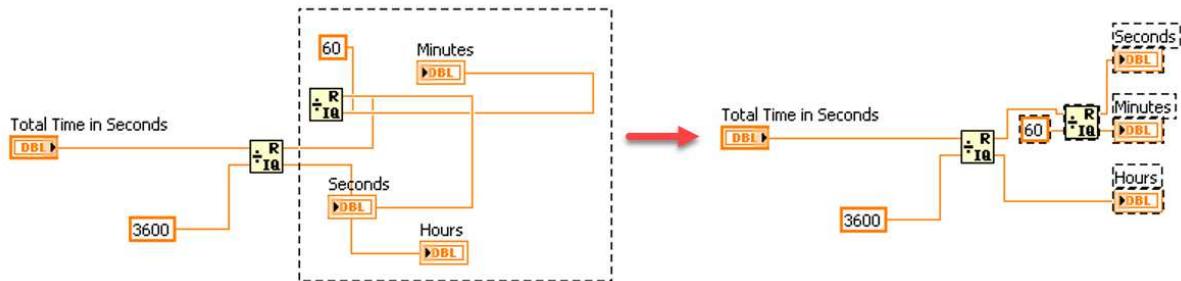
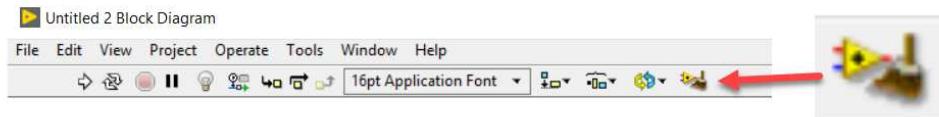
ในการเชื่อมต่อ Data คละชนิดกัน จะทำให้เกิด Broken Wire โดยจะมีเครื่องหมาย กากบาท อุ่ ตรงกลางและดังรูป



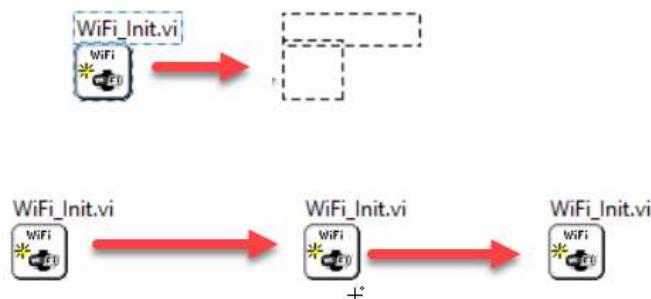
เมื่อจะใช้ฟังก์ชันใน LabVIEW มีจำนวนมาก คำอธิบายแต่ละฟังก์ชันสามารถเปิดการใช้งานโดยเลือก Help > Show Context Help จาก LabVIEW หรือกดคีย์ลัด Ctrl+H เป็นการเปิด/ปิด Context Help เป็นหน้าต่างที่อธิบายความหมายของฟังก์ชันต่างๆในโปรแกรม เมื่อนำ Cursor ไปไว้บน Nodes หรือฟังก์ชันใดๆ ในโปรแกรมจะแสดง Context Help อัตโนมัติ



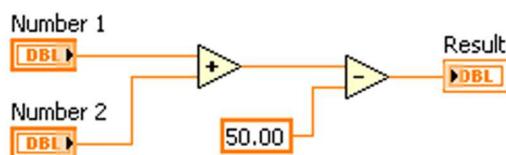
การเรียกใช้ฟังก์ชันใน LabVIEW ปกติจะทำได้โดย คลิกขวา จากนั้นเลือกฟังก์ชันและดังรูป แต่เมื่อการเรียกใช้ฟังก์ชันอีกครึ่งหนึ่งคือการ Search จากชื่อของฟังก์ชัน โดยคลิกที่เมนูบนขวาเลือก Search วิธีนี้นิยมใช้เฉพาะว่าทำให้การเขียนโปรแกรมได้เร็วขึ้น



สำหรับผู้เริ่มต้นใช้งาน LabVIEW มักพบปัญหาเดียวกันคือ โปรแกรมไม่เป็นระเบียบ เนื่องจากยังไม่คุ้นเคยกับการ Wiring โดยที่มุมบนขวาของ Block Diagram จะปุ่มชื่อว่า “Clean-Up Diagram” จะเป็นการจัดระเบียบโค้ดให้อัตโนมัติ โดยมีความสามารถจัดระเบียบเฉพาะส่วน โดยให้เลือกส่วนที่ต้องการ จากนั้นคลิกที่ “Clean-Up Diagram” ॥แสดงดังรูป



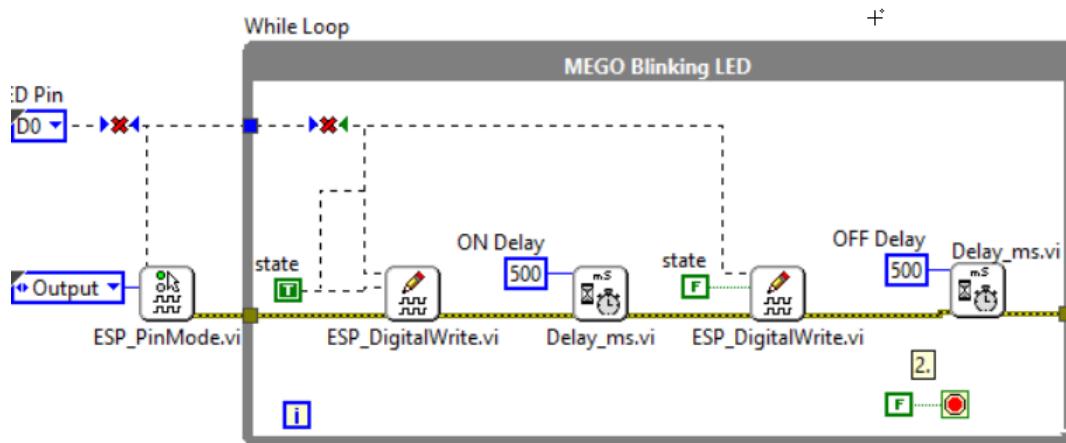
การเรียกใช้ฟังก์ชันเดิมบ่อยๆ สามารถทำการ Copy ฟังก์ชันใน LabVIEW ได้ โดยใช้คำสั่ง Copy (Ctrl+C) และ Paste (Ctrl+V) หรืออีกวิธีหนึ่งเรียกว่า Clone ทำได้โดย กดปุ่ม Ctrl ที่คีย์บอร์ดค้างไว้ จากนั้นคลิก+ลาก ฟังก์ชันที่ต้องการไปยังพื้นที่ที่ต้องการได้เลย



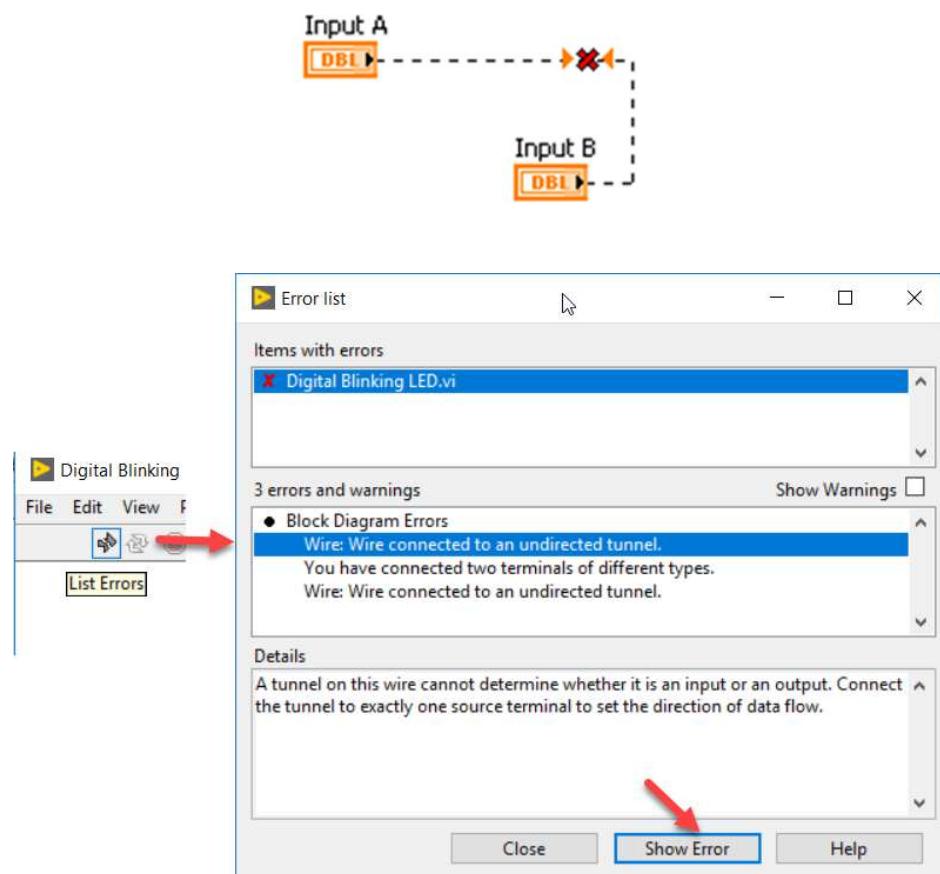
Data Flow เป็นหลักการทำงานของ LabVIEW ซึ่งเป็นวิธีการที่กำหนดว่าฟังก์ชันใดทำงานก่อน หรือหลัง โดยปกติภาษา Text Program อื่นๆ จะทำงานแบบ Top Down จากบนลงล่าง แต่ในภาษา LabVIEW เรียกว่า Data Flow คือปกติจะทำงานจากซ้ายไปขวา

การทำงานแบบ Dataflow คล้ายๆกับบ้าไอลจากที่สูงลงที่ต่ำ หรือจากซ้ายไปขวา โดยมีเงื่อนไข 2 ข้อ ดังต่อไปนี้ คือ

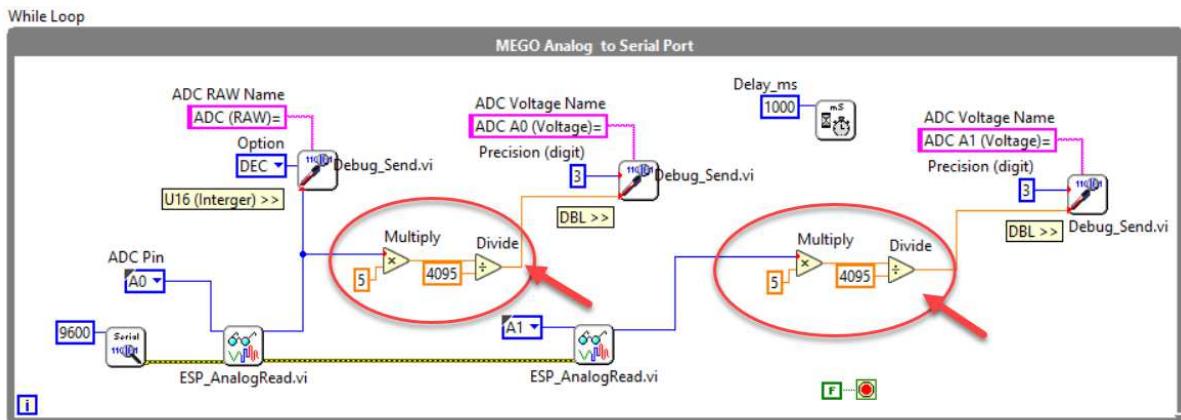
1. Node ต่างๆจะทำงานก็ต่อเมื่อ มี Data เข้ามาที่ Input ก็จะหมด (\*Required Input)
2. Node จะส่งค่าออกมา ก็ต่อเมื่อทำงานฟังก์ชัน นั้นๆใน Node เสร็จสมบูรณ์แล้ว (\*ถ้ายังทำงานไม่เสร็จ โปรแกรมจะค้างอยู่ที่ Node นั้นๆ)



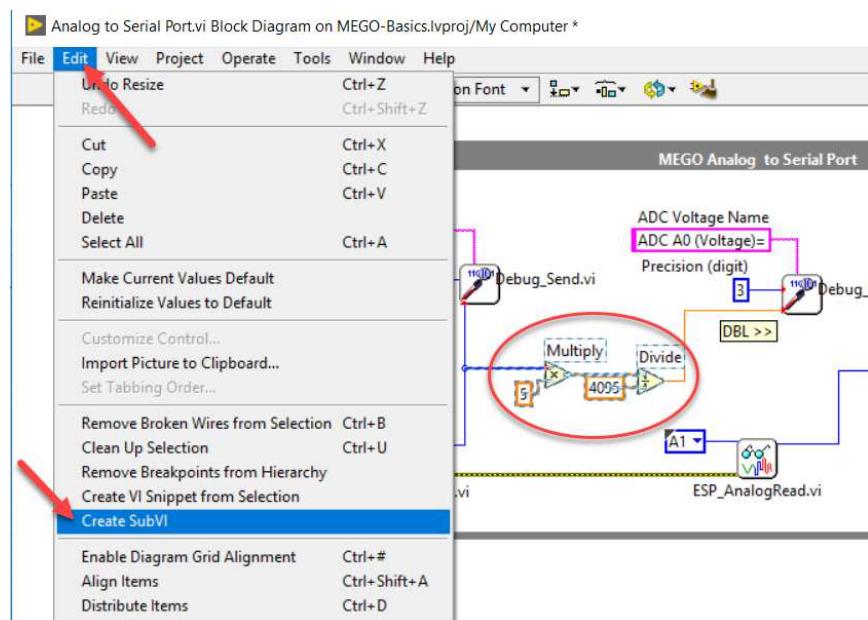
ในการนี้ถ้าหากเส้นเชื่อมต่อ Node ผิดพลาด ส่วนใหญ่เกิดจากลากเมาส์ Input<->Input, Output<->Output หรือเชื่อมต่อข้อมูลคนละชนิดกัน อาทิ ตัวแปลง Boolean เชื่อมต่อกับ Integer เป็นต้น ด้วยเหตุนี้จะทำให้เส้น Wire เป็นเส้นประ ซึ่งจะทำให้โปรแกรมมีปัญหาไม่สามารถทำงานได้



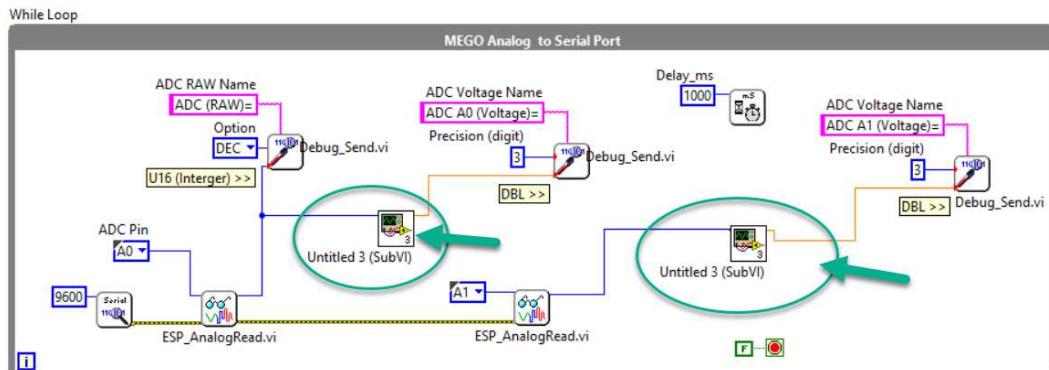
การตรวจสอบความผิดพลาดของโปรแกรม ทำได้โดยตรวจสอบจากปุ่ม Run ในกรณีที่โปรแกรมมีปัญหาจะมีลักษณะดังรูป เมื่อกดปุ่ม Run จะแสดงหน้าต่าง Error List สามารถกด Show Error จะเป็นการแสดงรายการ Error ที่จุดต่างๆ ของโปรแกรม ซึ่งง่ายกับการแก้ไขโปรแกรม

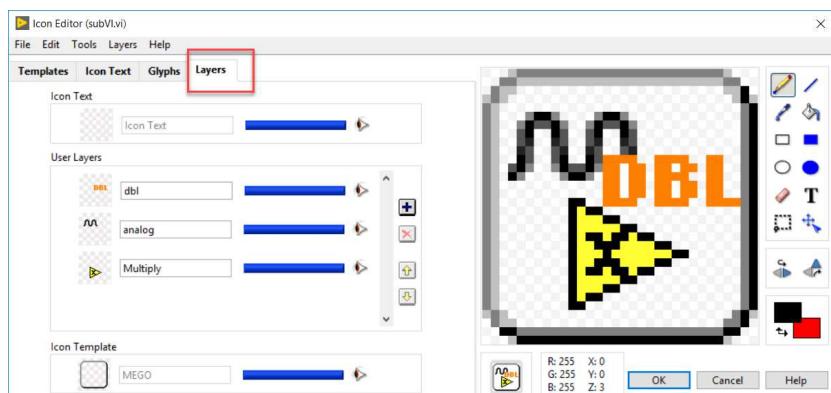
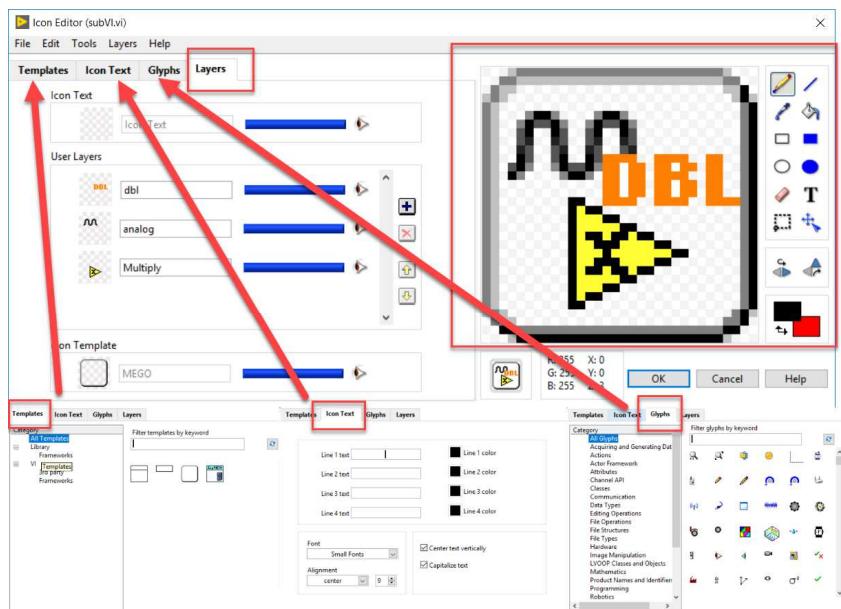
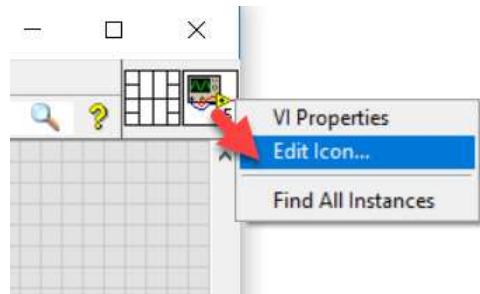


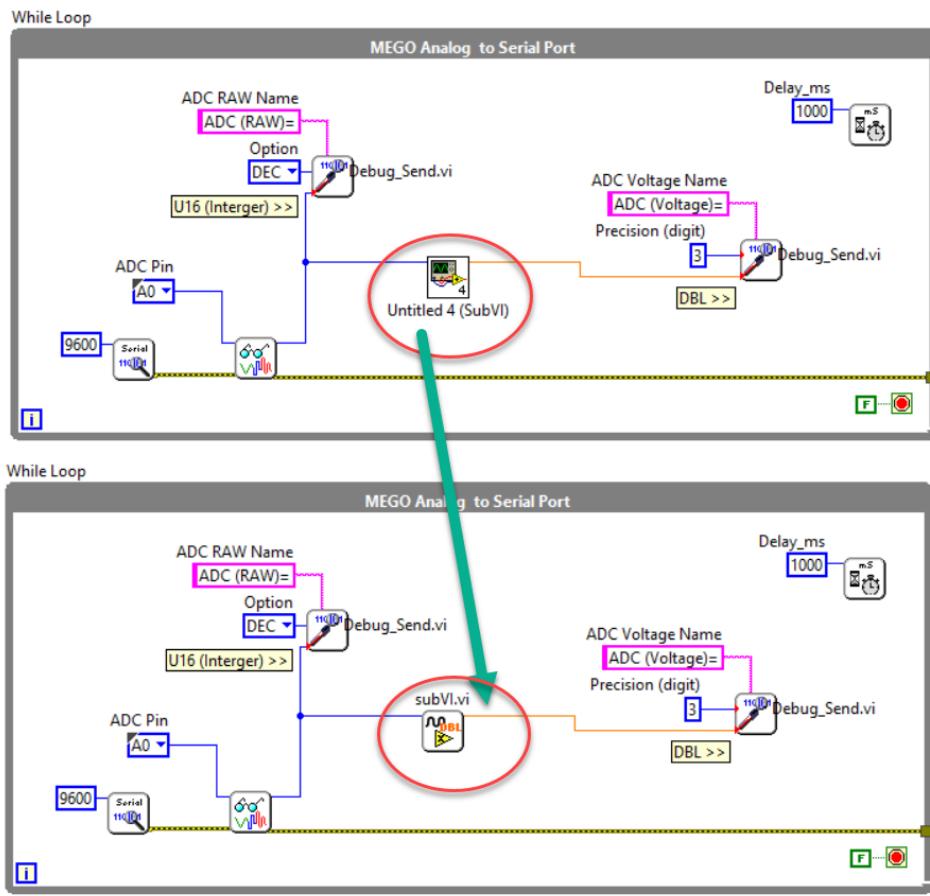
การเขียนโปรแกรมนักเรียนใช้งานฟังก์ชันที่ซ้ำซ้อน หรือมีโค้ดลักษณะที่คล้ายกัน การเขียนโปรแกรมที่ต้องรับฟังก์ชันนั้นๆ เป็นฟังก์ชันย่อย (SubVI) เนื่องจากจะทำให้เก็บโปรแกรมได้ง่ายในการหลัง



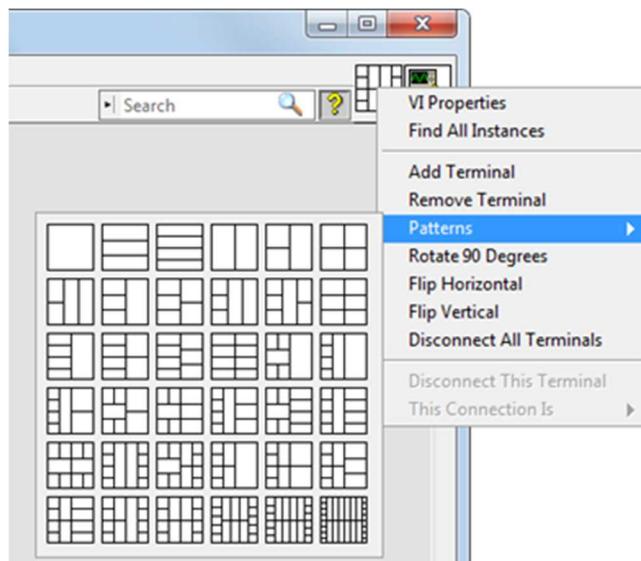
การสร้างโปรแกรมย่อย Sub-VI ทำให้โดย เลือกโค้ดในส่วนที่ต้องการ จากนั้นเลือก Edit > Create SubVI จากนั้นจะได้ SubVI อัตโนมัติ จากนั้นให้เปิด SubVI นั้นๆ แล้ว Save เพื่อบันทึกเป็นไฟล์ .vi เพื่อเรียกใช้งานได้ในภายหลัง



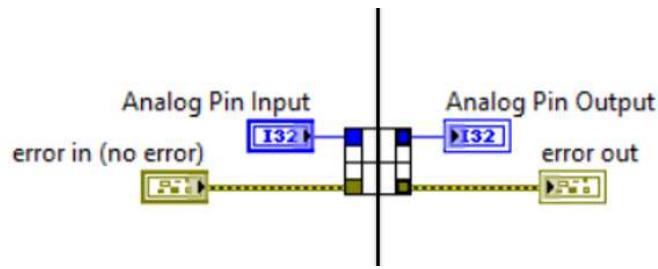




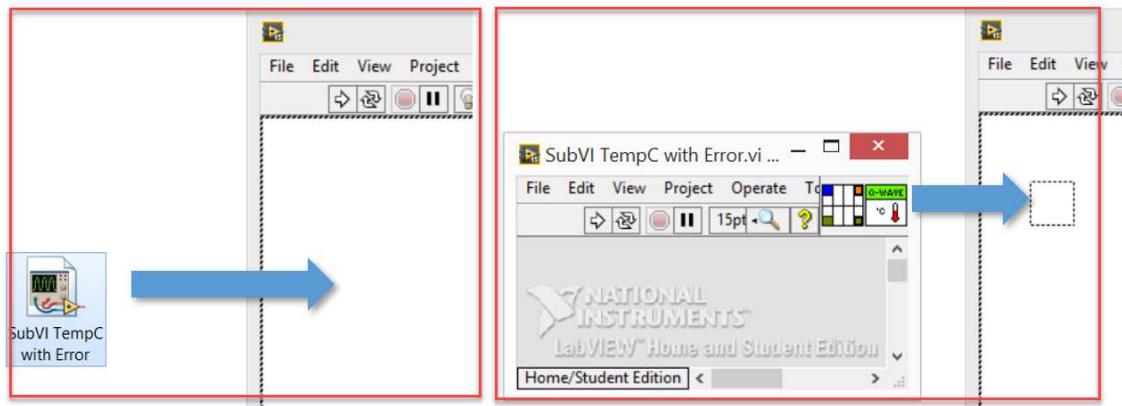
การสร้าง SubVI ที่ดีจะต้องออกแบบ Icon ให้สื่อความหมายของฟังก์ชันนั้นๆ ทำได้โดยคลิกขวาที่ Icon ด้านบนของ Front Panel เลือก Edit Icon ตัวอย่างของ SubVI ที่สร้าง Icon แล้วแสดงดังรูป



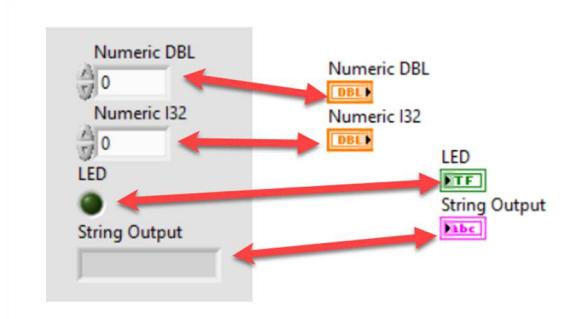
เมื่อสร้าง Icon แล้วขันตอนต่อไปต้องกำหนดจุดเชื่อมต่อ Connector ของ SubVI โดยคลิกขวาที่ บุมบนขวาที่จุด Terminal เลือก Patterns จากนั้นเลือกรูปแบบที่ต้องการตามจำนวนของ Connector ที่ต้องการใช้งาน



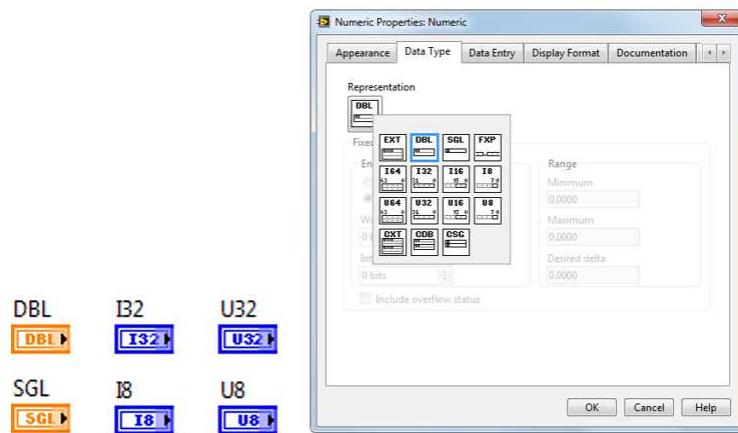
การกำหนดจุดเชื่อมต่อ โดยปกติถ้าเป็น Input จะกำหนดไว้ด้านซ้าย ถ้าเป็น Output จะกำหนดไว้ด้านขวา เพื่อการใช้งานสะดวกและเข้าใจง่าย



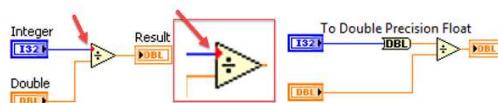
การใช้งาน SubVI ทำได้โดยลากมาใส่ที่ Block Diagram ได้เลย หรือในกรณีที่เปิด SubVI อยู่ให้คลิกที่รูป Icon และลากมาใส่ใน VI ที่ต้องการเรียกใช้งานได้เลย



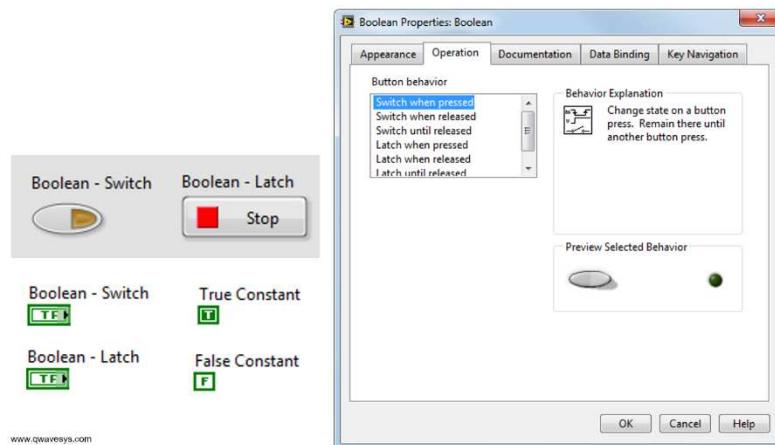
การสร้าง Control/Indicator ในหน้าต่าง Front Panel จะเป็นการสร้าง Terminal ใน Block Diagram ที่เชื่อมโยงกันให้อัตโนมัติ สำหรับบล็อกพัฒนาจะเขียนโปรแกรมในส่วน Block Diagram สำหรับ Front Panel จะเป็นส่วนที่แสดงผลสำหรับผู้ใช้งานเท่านั้น



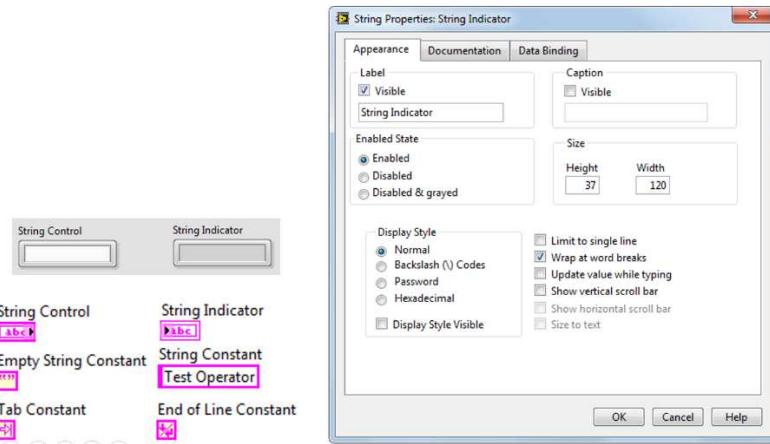
ตัวแปรชนิดตัวเลข (Numeric) จะมีสีสันและสีหน้าเงิน จะมี Data Type หลายรูปแบบ อาทิ Floating-point "DBL" (ทศนิยม), Unsigned integers "U16,U32" (จำนวนเต็ม ไม่มีเครื่องหมาย) และ Signed integers "I8,I16,I32" (จำนวนเต็ม มีเครื่องหมาย) การเปลี่ยน Data Type ทำได้โดยคลิกขวาที่ Numeric เลือก Property จากนั้นไปที่หน้าต่าง Data Type เพื่อกำหนดรูปแบบข้อมูล (Representation)



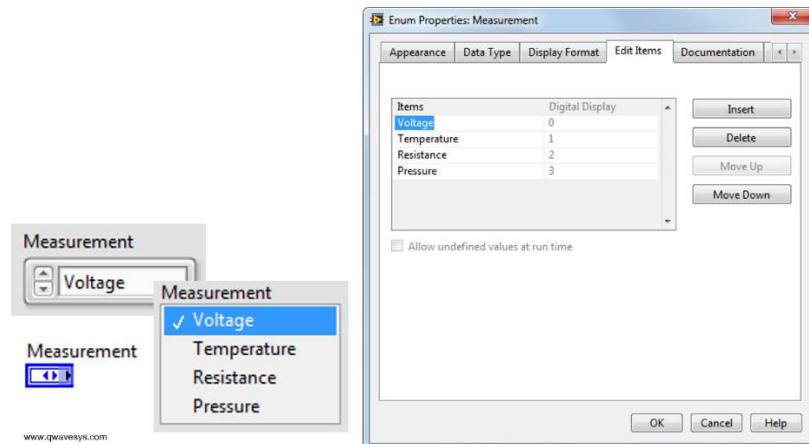
การเชื่อมต่อข้อมูลที่มี Data Type ต่างกันจะเกิดจุดสีแดงจะจังเตือนตรง Terminal นั้นๆ เรียกว่า "Coercion Dots" เป็นการแจ้งเตือนก่อนการเชื่อมต่อตัวแปรชนิดเดียวกัน เช่น Numeric ที่ต่างรูปแบบกัน (Data Type ก้าวเป็นชั้นนี้ โปรแกรมยังสามารถใช้งานได้ปกติ โดย LabVIEW จะแปลงข้อมูลให้อัตโนมัติ แต่ก็แม้ว่าจะใช้ได้ แต่ควรหลีกเลี่ยงให้เกิดขึ้น ควรเรียกใช้ฟังก์ชันแปลงตัวเลข เช่น "To Double Precision Float" แสดงดังรูป



ตัวแปรชนิดบูลีน (Boolean) จะมีสีเขียว เป็นตัวแปรที่มีสองสถานะ คือ True/False หรือ ON/OFF การใช้งาน Boolean ในหน้าต่าง Front Panel จะเน้นการสร้างสวิตช์ สามารถกำหนดลักษณะการทำงานได้ด้วย อาทิ สวิตช์เปิดปิด สวิตช์กดติดปล่อยดับ เป็นต้น การตั้งค่าในส่วนนี้เรียกว่า "Mechanical Action" ทำได้โดยคลิกขวาที่ Boolean นั้นๆ เลือก Mechanical Action ॥แสดงดังรูป

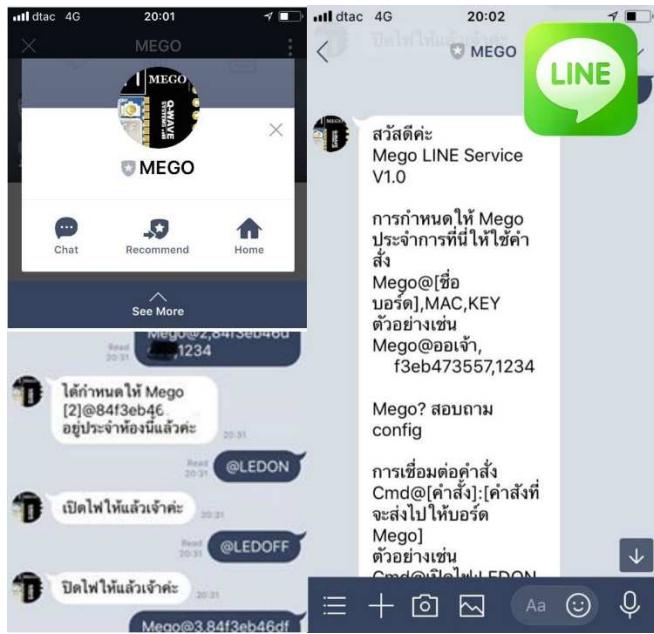


ตัวแปรตัวอักขร (String) จะมีสีชมพู เป็นข้อความเป็นผลรวมของตัวอักขร (ASCII) มักนิยมใช้งานเพื่อรับค่าข้อความจากผู้ใช้งานหรือ แสดงผลที่เป็นข้อความ สามารถกำหนดคุณสมบัติได้หลายประการ แสดงดังรูป

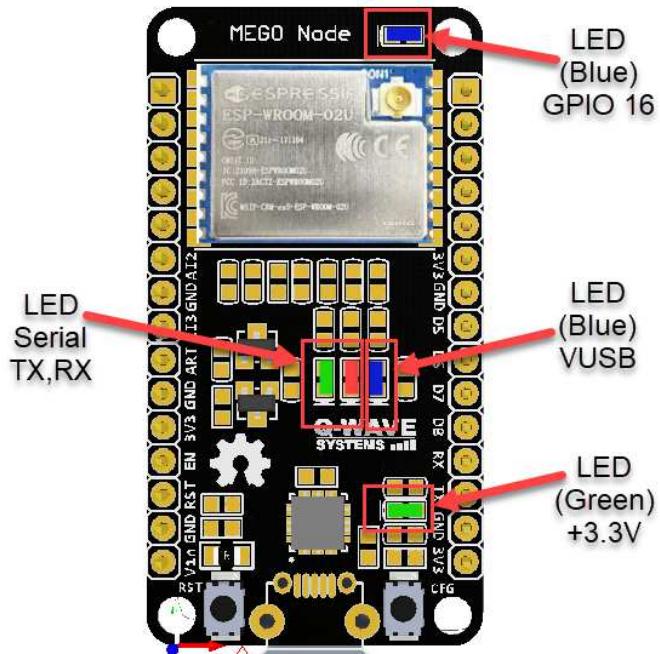


ตัวแปรชนิดอีนัม (Enum) เป็นตัวแปรที่ผู้ใช้กำหนดขึ้นมาเอง คือกลุ่มของตัวแปรชนิดตัวเลข (Integer) ที่แสดงผลเป็นข้อความ เพื่อให้ผู้ใช้เข้าใจง่าย และสะดวกในการใช้งาน ตัวอย่างตัวแปรที่ใช้ใน Enum อาทิ String 16-bit Integer เป็นต้น จากรูปจะเห็นว่าค่าตัวแปร = 0 จะมีชื่อว่า Voltage และค่าตัวแปร = 1 จะมีชื่อว่า Temperature เป็นต้น

## การใช้งานบอร์ด MEGO Node ร่วมกับ MEGO LINE Service

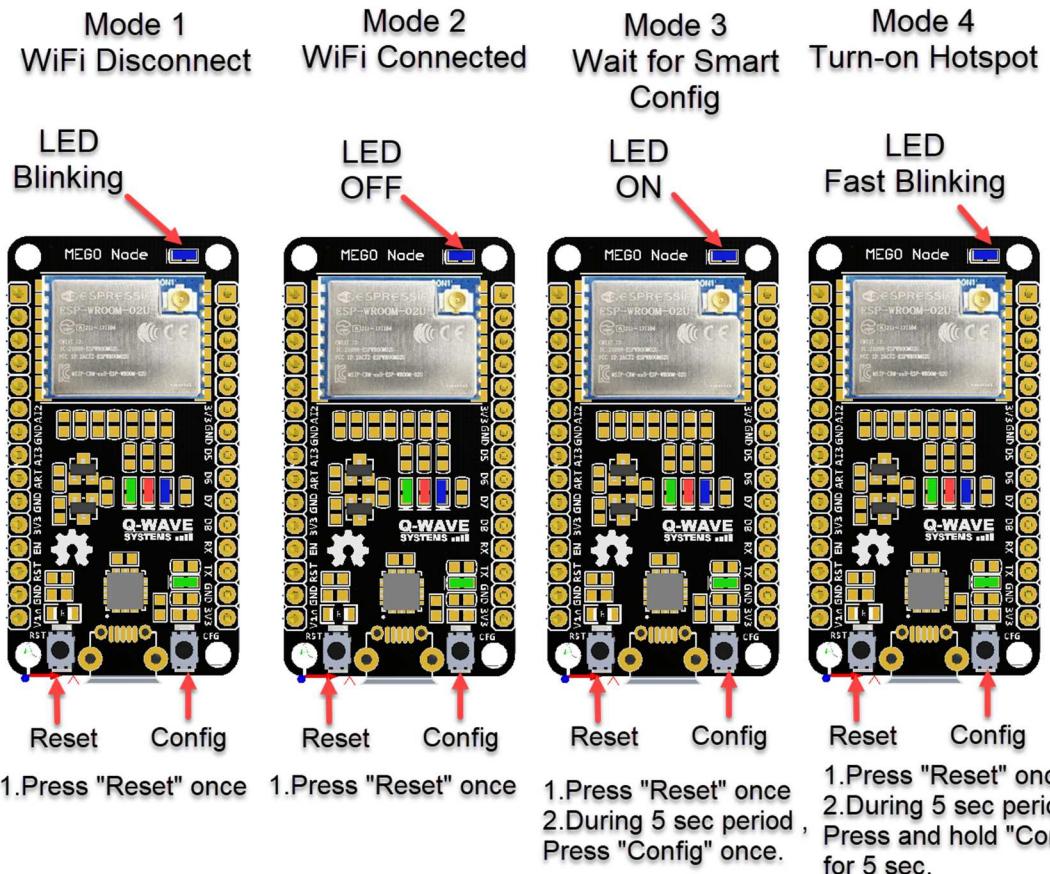


1.บอร์ด MEGO Node จะมีโปรแกรมตัวอย่างการเชื่อมต่อกับ MEGO LINE Service มาให้เรียบร้อยร้อย เมื่อเสียบสาย USB จะเห็นว่า LED สีฟ้า (GPIO16) ติดกระพริบ แสดงว่าบอร์ดอยู่ในสถานะ Mode 1 (WiFi Disconnect) คือพยายามเชื่อมต่อ WiFi แต่ไม่สามารถเชื่อมต่อได้



2.คำอธิบายโหมดการทำงานของโปรแกรม ของบอร์ด MEGO Node จะมี 4 สถานะดังนี้

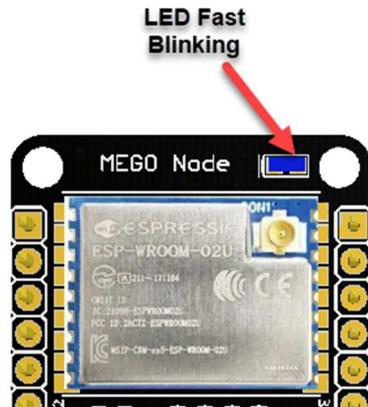
# LED GPIO16 Status Indicator



Mode 1 = WiFi Disconnect គឺបែវចិត្តពួយការមិនចូលរួមទៅ WiFi ព័ត៌មានសារកណ្តុះស្រាវជ្រាវ។

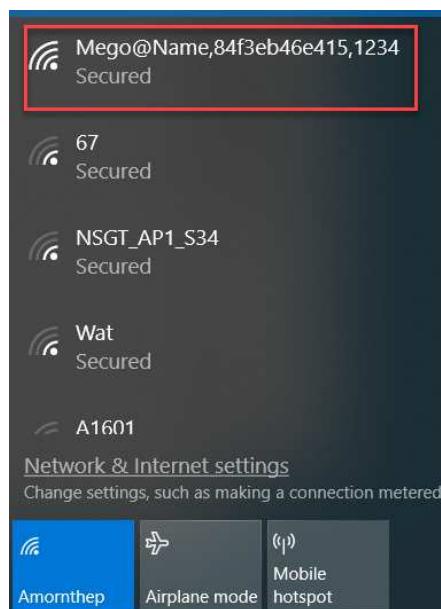
Mode 4 = บอร์ดจะเปิด WiFi ของตัวเอง โดยกำกับตัวเองเป็น Hotspot ที่มีชื่อดังนี้  
“Mego@Name,[MAC ADDRESS],[Default Password]” ตัวอย่างเช่น  
“Mego,Name,85fg63dj6612,1234” โดยผู้ใช้งานสามารถแก้ไขได้จากโปรแกรมภายหลัง

3.ตั้งค่าให้บอร์ดทำงานใน Mode 4 (Turn-ON WiFi Hotspot) เพื่อจะดูค่า MAC ADDRESS ของบอร์ด ทำได้โดย กดปุ่ม Reset 1 ครั้ง จากนั้นภายใน 5 วินาที ให้กดปุ่ม Config ค้างไว้อย่างน้อย 5 วินาที จะเห็นว่า LED GPIO16 จะกระพริบเร็วขึ้น และแสดงว่าบอร์ดอยู่ในสถานะ Hotspot = Turn ON



ณ ตอนนี้บอร์ด MEGO จะทำการปล่อยสัญญาณ WiFi ที่มีชื่อลักษณะนี้ “Mego@Name,[MAC ADDRESS],[Default Password]”

ขั้นตอนถัดไป เข้าไปที่คอมพิวเตอร์ หรือ Smart Phone เปิดดูสัญญาณ WiFi จะพบบอร์ด MEGO แสดงดังรูป



\*ให้ทำการจดบันทึก “MAC ADDRESS” ของบอร์ดเอาไว้ เนื่องจากว่าต้องใช้ในการ Register เข้าระบบเพื่อใช้งานกับ LINE Service ในภายหลัง

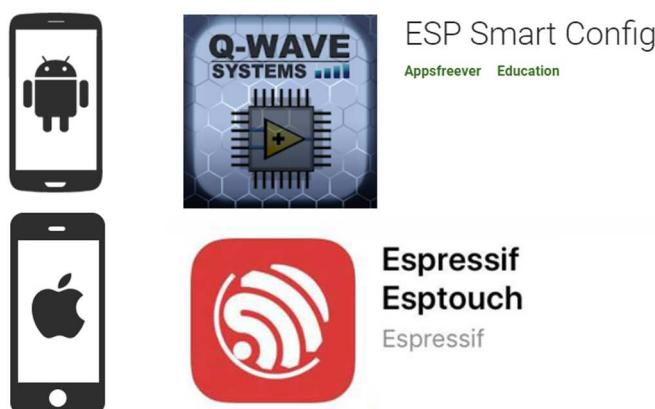
4.ตั้งค่าให้บอร์ดทำงานใน Mode 3 (Wait for Smart Config) ทำได้โดย กดปุ่ม Reset 1 ครั้ง จากนั้น ภายใน 5 วินาที ให้กดปุ่ม Config 1 ครั้ง จะเห็นว่า LED GPIO16 จะติดค้าง แสดงว่าบอร์ดอยู่ในสถานะ Wait for Smart Config



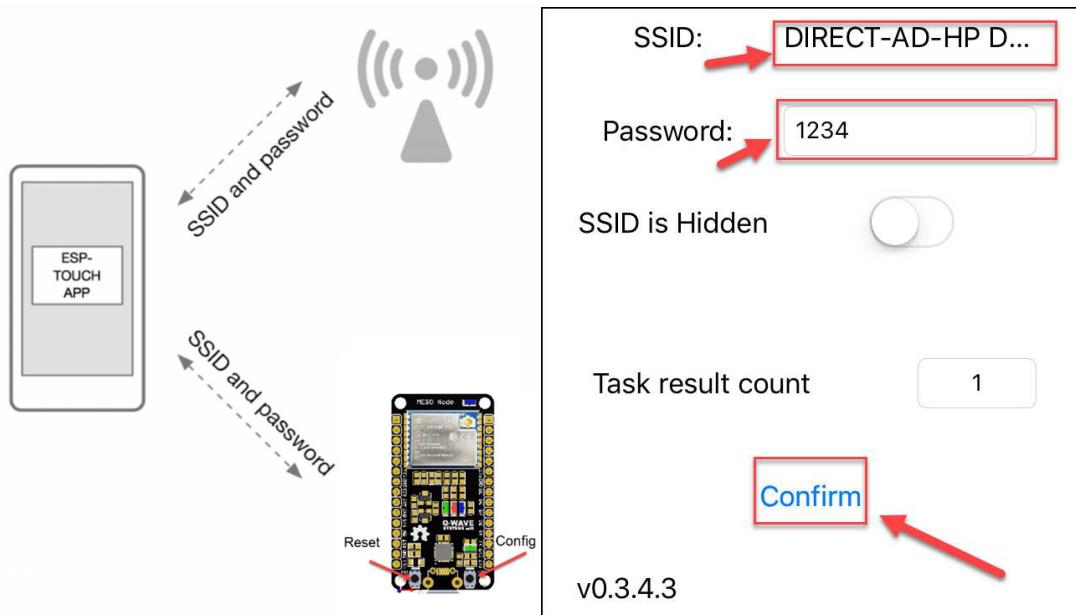
5.ตั้งค่าให้บอร์ด MEGO Node เชื่อมต่ออินเทอร์เน็ต WiFi ได้นั้น จะต้องทำผ่าน App จาก Smart Phone โดยใช้ Apple iOS หรือ Android ก็ได้ โดยต้องดาวน์โหลดแอพในมือถือดังนี้

สำหรับ Android โหลดแอพชื่อว่า “ESP Smart Config”

สำหรับ Apple iOS โหลดแอพชื่อว่า “Espressif Esptouch” (Esptouch)

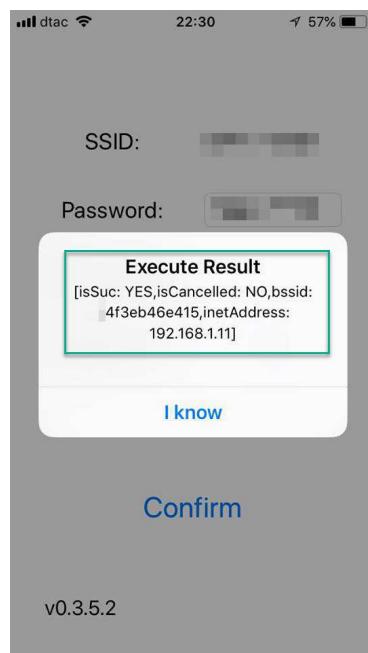


หลักการทำงานคือ App จะส่ง SSID และ Password ของ WiFi กี่ต้องการเชื่อมต่อ เข้าไปผังตัวกับบอร์ด MEGO Node ให้อัตโนมัติ การใช้งานเพียงแค่กรอกข้อมูล WiFi จากนั้นกด “Confirm” แสดงดังรูป



จากบัน្តօសັກຮູ່ App ຈະແສດງผลການກຳຈານເສັ້ນສົມບູນ ຈະແສດງ MAC ADDRESS ແລະ IP ADDRESS ຂອງບວດທີ່ຕັ້ງຄ່າເຮັຍບ້ອຍແລ້ວ

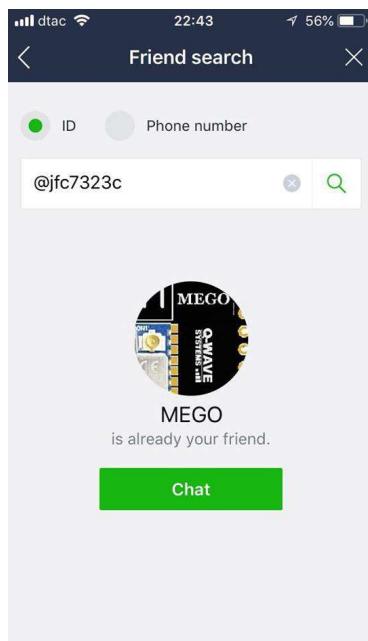
ສັງເກດ LED GPIO16 ກົ່ວບວດ MEGO ຈະດັບ (OFF) ແລະ ວ່າມຕົວກັບ WiFi ໄດ້ເຮັຍບ້ອຍແລ້ວ



\*ການໃຊ້ງານ Smart Config ສາມາດຕັ້ງຄ່າໄຟກັບບວດໄດ້ພຽມກັນ ມາກກວ່າ 1 ບວດ ອາຖິ ດ້ວຍ  
ບວດ MEGO Node 10 ບວດ ກີ່ກຳໄຟແຕ່ລະບວດເຂົ້າ Mode 3 (Wait for Smart Config) ຮອໃວ ຈາກນັ້ນ  
ໃຊ້ App ຕັ້ງຄ່າໄຟເຍັວທັງ 10 ບວດໄດ້ເລີຍ

6. เชื่อมต่อ MEGO เข้ากับ LINE Service กึ่งขั้นตอนนี้ แสดงว่าบอร์ด MEGO พร้อมเชื่อมต่อ กับ LINE แล้ว ที่ Smart Phone เข้าเข้า App LINE จากนั้นเพิ่มเพื่อน โดยการค้นหาจาก “LINE ID” ดังนี้

LINE ID:  
“@jfc7323c” หรือ “@ntg7407h”



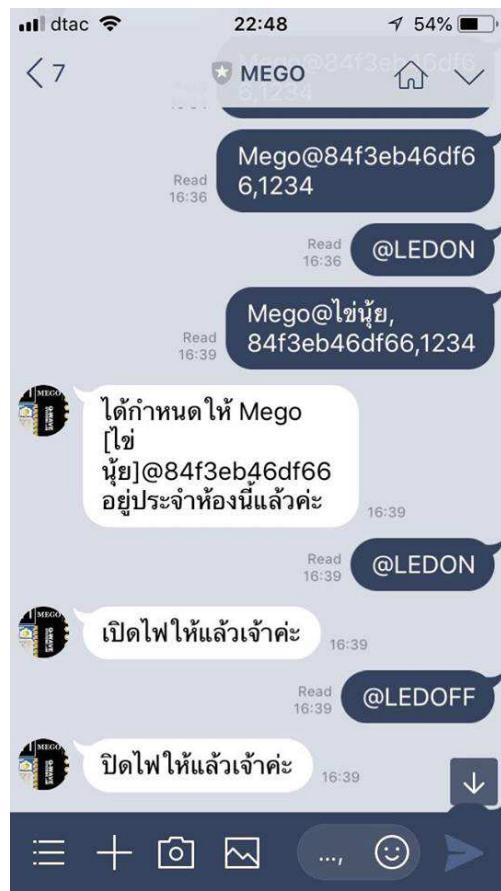
เมื่อเพิ่มเพื่อน MEGO เรียบร้อยแล้ว ให้ทำการ Register บอร์ด MEGO เพื่อใช้งาน โดยการพิมพ์คำสั่งดังนี้

“Mego@[ชื่อบอร์ดที่ต้องการตั้งชื่อ],[MAC ADDRESS],[Default Password]”

เช่น

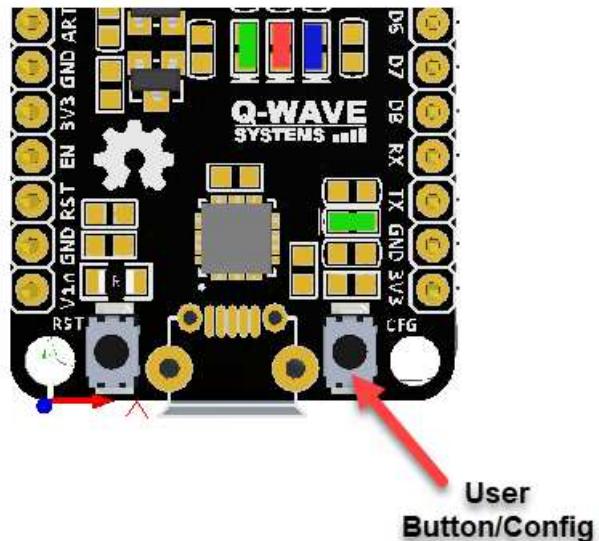
“Mego@**ไทย**,**84f3eg66gg22**,**1234**”

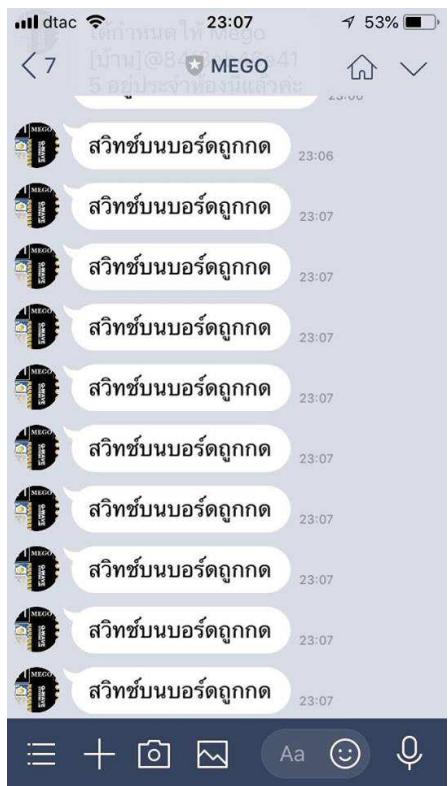
\*เนื่องจากว่า Default Password กำหนดจากโปรแกรมในบอร์ด  
ดังนั้นต้องกำหนดเป็น 1234 เท่านั้น โดยโปรแกรมนี้มีจุดประสงค์เพื่อ Demo ทดสอบการเชื่อมต่อ บอร์ด MEGO กับ LINE Service เท่านั้น สำหรับการใช้งานจริง บังพันสามารถกำหนด Password ได้อ่องในภายหลัง



จากนั้นกดลงใช้คำสั่ง “@LEDOFF” จะเห็นว่า LED GPIO16 ที่บอร์ดจะดับ และคำสั่ง “@LEDON” จะเห็นว่า LED GPIO16 จะติด

ขั้นตอนต่อไปให้กดลง กดปุ่ม “Config” บนบอร์ด MEGO จะเห็นว่าใน LINE จะแจ้งเตือนว่ามีการกดปุ่มบนบอร์ด





ณ ตอนนี้เราสามารถควบคุมบอร์ด MEGO ผ่าน LINE Service ได้เรียบร้อยแล้ว โดยคำสั่งเพิ่มเติม เพื่อใช้งาน ให้กดลงดังนี้

คำสั่ง “?” เป็นคุณมือการใช้งานต่าง ๆ



คำสั่ง “Demo?” เพื่อแสดงคำสั่งที่ใช้งานได้เพิ่มเติม



คำสั่ง “Cmd@[คำสั่งที่ต้องการ]:[คำสั่งที่จะส่งไปให้บอร์ด]” เพื่อเป็นการเชื่อมโยงคำสั่งที่ผู้ใช้งานสามารถกดตั้งค่าได้เอง เช่น “Cmd@เปิดไฟ:LEDON” ซึ่งจะทำการเชื่อมโยงคำว่า “เปิดไฟ” และ “LEDON” เข้าด้วยกัน การใช้งานสามารถสั่งได้ดังนี้ “[ชื่อบอร์ด][คำสั่ง]” เช่น “ไปบุญเปิดไฟ” จะสเมือนว่าเราสั่งว่า “@LEDON”