เรียนรู้ เข้าใจ ใช้งาน
BBC micro:bit ด้วยภาษา MicroPython

ดนัย เจษฎาฐิติกุล
จารุต บุศราทิจ

ETT
บริษัท อีทีที จำกัด
1112/96-98 ถนนสุขุมวิท แขวงพระโขนง เขตคลองเตย กรุงเทพฯ 10110
1112/96-98 Sukhumvit Rd., Phrakanong Klongtoey Bangkok 10110
Tel : 02-7121120 Fax : 02-3917216
http://www.etteam.com
LINE ID : etteam1
f : facebook.com/ett.co.th
e-mail : sale@etteam.com

คำนำ

จากเทคโนโลยีในปัจจุบันทำให้การศึกษาด้านไมโครคอนโทรลเลอร์สามารถเข้าถึงได้ง่าย โดยมีไมโครคอนโทรลเลอร์ที่ประสิทธิภาพสูงในราคาที่ประหยัด และถูกนำมาใช้ในการเรียนการสอนตั้งแต่ในระดับประถมศึกษา ประกอบกับบริษัทยักษ์ใหญ่ด้านการแพร่ภาพและกระจายเสียงอย่าง BBC ได้ผลิตบอร์ดไมโครคอนโทรลเลอร์ขนาดเล็กเพื่อแจกจ่ายให้กับนักเรียนโดยได้รับความร่วมมือจากองค์กรต่าง ๆ ได้แก่ บริษัทไมโครซอฟท์ มหาวิทยาลัยแลนแคสเตอร์ บริษัทอีลิเม็น 14 และบริษัทซัมซุง ผลิตบอร์ดไมโครคอนโทรลเลอร์ชื่อ micro:bit ซึ่งใช้ไมโครคอนโทรลเลอร์สถาปัตยกรรม 32 บิต ตระกูล ARM Cortex-M0 เป็นตัวประมวลผลหลัก พร้อมทั้งชุดวงจรพื้นฐานบนบอร์ด เช่น สวิตช์ เมตริกซ์แอลอีดี เซ็นเซอร์อุณหภูมิ เซ็นเซอร์เข็มทิศ เซ็นเซอร์การเคลื่อนไหว ระบบสื่อสารคลื่นวิทยุ เป็นต้น และชุดไลบรารีทางด้านการสังเคราะห์เสียง/แปลงข้อความเป็นเสียง/เล่นเสียงตามโฟนีม ทำให้สามารถประยุกต์ใช้บอร์ด micro:bit กับการเรียนการสอน หรือนำไปประยุกต์ใช้เพื่อสร้างชิ้นงานได้สะดวก และสามารถเลือกเรียนรู้การเขียนโปรแกรมควบคุมได้หลากหลายภาษา เช่น ภาษาบล็อก จาวาสคริปต์ หรือภาษาไพธอน โดยแต่ละภาษามีจุดดีจุดเด่นที่แตกต่างกันไป เช่น ภาษาบล็อกเหมาะสมกับการเรียนรู้การเขียน/สั่งการด้วยภาพทำให้ง่ายต่อการเข้าถึง ภาษาจาวาสคริปต์เป็นการเขียนภาษาที่ไม่ซับซ้อนใช้ทรัพยากรของบอร์ดน้อย และภาษาไพธอนรองรับการเขียนโปรแกรมได้หลากหลายขึ้นแต่สิ้นเปลืองทรัพยากรของบอร์ดไมโครคอนโทรลเลอร์มากขึ้น

หนังสือเล่มนี้เขียนขึ้นโดยใช้ภาษาไพธอนเป็นภาษาสำหรับควบคุมการทำงานของบอร์ด BBC micro:bit เพื่อให้ผู้อ่านตั้งแต่ระดับมัธยมศึกษาสามารถอ่านเพิ่มเติมเพื่อเสริมความรู้ หรือนำไปปรับปรุงโปรแกรมสำหรับใช้งานในระดับที่ซับซ้อนยิ่งขึ้น ซึ่งหนังสือแบ่งเป็น 3 เนื้อหาหลัก คือ

- ส่วนของการใช้งานเครื่องมือและภาษาไพธอน ที่อธิบายถึงวิธีการใช้เครื่องมือ Thonny และ Mu สำหรับเขียนโปรแกรมภาษาไพธอนแบบออฟไลน์ หรือไม่ต้องเขียนผ่านอินเทอร์เน็ต หลังจากนั้นอธิบายหลักภาษา พร้อมตัวอย่างการเขียนและคำอธิบาย และปิดท้ายด้วยเรื่องของคลังไลบรารีที่ควรรู้จักและ

มีประโยชน์ต่อการนำไปประยุกต์เขียนโปรแกรมภาษาไพธอนต่อไป โดยเน้นเฉพาะไลบรารีและชุดคำสั่งที่ทำงานได้กับบอร์ด BBC micro:bit

- ส่วนของการเขียนภาษาไพธอนเพื่อเชื่อมต่อ/ควบคุมการทำงานของอุปกรณ์ภายนอก เช่น แอลอีดี แอลอีดีอาร์จีบี (RGB LED) เมตริกซ์แอลอีดี สวิตช์กดติดปล่อยดับ สวิตช์แม่เหล็ก มอเตอร์ อัลตราโซนิก อินฟราเรด เซ็นเซอร์อุณหภูมิ เซ็นเซอร์ความชื้น ตัวต้านทานเปลี่ยนค่าได้ แอลซีดีตัวอักษร รีเลย์ หน่วยความจำ EEPROM ลำโพงขนาดเล็ก เซ็นเซอร์ความชื้นภายในดิน เซ็นเซอร์ตรวจจับความเคลื่อนไหว ไอซีขยายพอร์ต นาฬิกาฐานเวลาจริง (RTC) เป็นต้น
- ส่วนของการประยุกต์ใช้เพื่อควบคุมการทำงานของหุ่นยนต์รถที่ใช้มอเตอร์ประเภทเซอร์โวในการขับเคลื่อนและมีเซ็นเซอร์วัดระยะทาง ตรวจจับเส้น และการชนกับวัตถุติดตั้งบนหุ่นยนต์ พร้อมอุปกรณ์ภายนอกเช่น LED RGB และลำโพงขนาดเล็กสำหรับใช้เป็นภาคแสดงผลการทำงาน เป็นต้น และเสริมเรื่องของการควบคุมหุ่นยนต์แบบไร้สายด้วยการสื่อสารสัญญาณวิทยุ อันเป็นคุณสมบัติที่มากับบอร์ด micro:bit

สุดท้าย หนังสือเล่มนี้คงสำเร็จเป็นรูปเล่มไม่ได้ถ้าขาดการสนับสนุนจากคุณกอบกิจ เติมผาติ และทีมงานบริษัท อีทีที จำกัด ที่ร่วมกันให้แนวทางและข้อมูลสำหรับการเขียนหนังสือ และครอบครัวที่ให้กำลังใจตลอดมา ทั้งนี้ ถ้าหนังสือมีข้อผิดพลาดประการใดผู้เขียนขออภัยล่วงหน้าและน้อมรับข้อแนะนำ/ข้อเสนอแนะเพื่อนำไปปรับปรุงและให้ดาวน์โหลดส่วนของปรับปรุงต่อไปในอนาคต

ดนัย เจษฎาฐิติกุล
จารุต บุศราทิจ
1 กันยายน 2563

สารบัญ

เนื้อหา	หน้า
บทที่ 1 แนะนำ MicroPython	**1**
แนะนำ BBC micro:bit	1
แนะนำ ET-BBC GPIO V1	3
แนะนำ MicroPython	6
ขั้นตอนการพัฒนาโปรแกรม	7
การใช้ Thonny	7
การติดตั้งโปรแกรม Thonny	8
ไอคอนสั่งงานของ Thonny	10
การเขียนโค้ดและรันโปรแกรมของ Thonny	11
การใช้ Mu Editor	15
สรุป	18
บทที่ 2 ภาษาไพธอน	**21**
ภาษาไพธอน	21
ตัวแปร	22
หลักการตั้งชื่อตัวแปร	23
การกำหนดค่าให้กับตัวแปร	24
การตรวจสอบประเภทของตัวแปร	25
การแปลงชนิดตัวแปร	25
ประเภทข้อมูล	25
ตัวเลข	26
ข้อความ	27
ลิสต์	33
ทูเพิล	39
ดิกชันนารี	40
การดำเนินการนิพจน์	42
ตัวดำเนินการคณิตศาสตร์	44

เนื้อหา	หน้า
ตัวดำเนินการเปรียบเทียบและตรรกะ	45
ตัวดำเนินการบิต	45
ตัวดำเนินการข้อความ	45
ลำดับในการดำเนินการ	47
ข้อคิดเห็นในโปรแกรม	47
เงื่อนไข	48
เงื่อนไขแบบ ถ้า...แล้ว	48
เงื่อนไขแบบ ถ้า...แล้ว...ไม่เช่นนั้น	50
เงื่อนไขแบบซ้อนหลายเงื่อนไข	51
ทำซ้ำ	52
การทำซ้ำในช่วงค่า	52
การทำซ้ำเมื่อเงื่อนไขเป็นจริง	54
การทำซ้ำด้วยชุดข้อมูล	57
การทำซ้ำหลายชั้น	58
คำสั่งควบคุมการทำซ้ำ	59
โปรแกรมย่อย	63
การสร้างและการใช้งานโปรแกรมย่อย	64
อาร์กิวเมนต์	64
การใส่ค่าอาร์กิวเมนต์แบบไม่จำกัดจำนวน	65
การคืนค่า	66
ขอบเขต	67
คลาส	69
การสร้างคลาส	69
การสืบทอดคลาส	70
การแทนที่เมธอด	72
สรุป	73
บทที่ 3 ไลบรารีของไมโครไพธอน	**75**
ไลบรารีมาตรฐาน	75
ไลบรารี sys	76
ไลบรารี gc	78
ไลบรารี math	79
ไลบรารี collections	82
ไลบรารี os	84

เนื้อหา	หน้า

ไลบรารี struct 85
ไลบรารี time 87
ไลบรารี random 89
ไลบรารี machine 92
ไลบรารี microbit 93
สรุป 95
บทที่ 4 การแสดงผล 97
การใช้คำสั่ง print 97
การแสดงผลแอลอีดี 99
การแสดงสั่งเปิดและปิดหลอดแอลอีดี 99
การแสดงตัวอักษร 102
การแสดงภาพ 103
การแสดงภาพที่สร้างเอง 104
การแสดงเอนิเมชันที่แอลอีดี 105
การพูด 108
เสียงดนตรี 111
เสียงเอฟเฟค 114
สรุป 115
บทที่ 5 การนำเข้า/ส่งออกและการสื่อสาร 117
พอร์ตของ micro:bit 117
การนำเข้าสัญญาณแอนาล็อก 119
การนำออกสัญญาณดิจิทัล 120
การนำออกสัญญาณแอนาล็อก 120
ปุ่มบนบอร์ด micro:bit 122
การสื่อสารอนุกรม 123
การนำเข้าข้อมูลจากพอร์ตสื่อสารอนุกรม 126
เทคนิคการตรวจสอบการแปลงข้อมูล 128
การสื่อสาร I^2C 131
การสื่อสาร SPI 134
การตรวจสอบเหตุการณ์จากการเคลื่อนที่ 135
เจสเชอร์ 137
เซ็นเซอร์เข็มทิศ 139
สรุป 140

เนื้อหา	หน้า
บทที่ 6 เซอร์โวมอเตอร์	**141**
เซอร์โวมอเตอร์	141
การสั่งงานเซอร์โวมอเตอร์	142
ตัวอย่างโปรแกรมซ้าย/กลาง/ขวา/กลาง	144
ตัวอย่างโปรแกรมหมุนซ้ายขวา	147
สรุป	148
บทที่ 7 การใช้ ET-RGB 1x1 Kit	**149**
แนะนำ ET-RGB1x1 Kit	149
ไลบรารี neopixel	151
ตัวอย่างแสดงหลอดสีแดง เขียว และเหลือง	152
ตัวอย่างเฟดสี	153
ตัวอย่างไฟวิ่งไล่เฉดสีแดง	154
สรุป	154
บทที่ 8 การใช้งาน HC-SR04 Ultrasonic Sensor	**155**
แนะนำ HC-SR04 Ultrasonic Sensor	155
วิธีการคำนวณหาระยะ	157
ตัวอย่างโปรแกรมอ่านและแสดงค่า	158
สรุป	160
บทที่ 9 การใช้งาน SENSOR SHT-31	**161**
แนะนำ ET-SENSOR SHT31	161
การทำงานของ SHT31	162
วงจรของ SHT31	163
การแปลงค่าความชื้น	164
การแปลงค่าอุณหภูมิ	164
ตัวอย่างอ่านค่าความชื้นและอุณหภูมิ	165
สรุป	166
บทที่ 10 การใช้งาน ET-MINI 24xx	**167**
แนะนำบอร์ด ET-MINI 24xx	167
วงจรของ ET-MINI 24xx	168
การเขียนข้อมูลลง EEPROM	170
การอ่านข้อมูลจาก EEPROM	170
ตัวอย่างโปรแกรมเขียนข้อมูลลง EEPROM	170
ตัวอย่างโปรแกรมอ่านข้อมูลจาก EEPROM	172

เนื้อหา	หน้า
สรุป	174
บทที่ 11 การใช้งาน R-TRACKER3	**175**
แนะนำบอร์ด R-TRACKER3	175
วงจรและการใช้งาน R-TRACKER3	176
ตัวอย่างโปรแกรมอ่านค่าเซ็นเซอร์	178
ตัวอย่างโปรแกรมตรวจสอบเซ็นเซอร์ซ้าย/กลาง/ขวา	179
สรุป	180
บทที่ 12 การใช้งาน ET-SW KIT	**181**
แนะนำบอร์ด ET-SW KIT	181
วงจรและการใช้งาน ET-SW KIT	182
ตัวอย่างโปรแกรมการรับข้อมูลจากสวิตช์	183
ตัวอย่างโปรแกรมการรับข้อมูลจากสวิตช์แบบดีเบาซ์	184
สรุป	186
บทที่ 13 การใช้งาน ET-RELAY KIT	**187**
แนะนำบอร์ด ET-RELAY KIT	187
วงจรและการใช้งาน ET-RELAY KIT	187
ตัวอย่างโปรแกรม	189
สรุป	190
บทที่ 14 การใช้งาน ET-BUZZER KIT	**191**
แนะนำ ET-BUZZER KIT	191
การวนรอบเพื่อสร้างความถี่	192
ตัวอย่างเล่นเสียงตัวโน้ต	196
สรุป	197
บทที่ 15 การใช้งาน ET-LED TRAFFIC (LED TRAFFIC) KIT	**199**
แนะนำ ET-LED TRAFFIC (LED TRAFFIC) KIT	199
วงจรและการใช้งาน	200
ตัวอย่างโปรแกรมขับหลอดแอลอีดี	201
ตัวอย่างโปรแกรมจำลองสัญญาณไฟจราจร	203
สรุป	204
บทที่ 16 การใช้งาน ET-LED TRAFFIC (LED RGB) KIT	**205**
แนะนำ ET-LED TRAFFIC (LED RGB) KIT	205
วงจรและการใช้งาน	206
ตัวอย่างโปรแกรมขับหลอดแอลอีดี	206

เนื้อหา	หน้า
ตัวอย่างโปรแกรมผสมสี	208
ตัวอย่างโปรแกรมจำลองสัญญาณไฟจราจร	209
สรุป	211
บทที่ 17 การใช้งาน ET-ADC VR-NTC (VR 10K) KIT	**213**
แนะนำบอร์ด ET-ADC VR-NTC (VR 10K) KIT	213
วงจรและการทำงานของบอร์ด ET-ADC VR-NTC (VR 10K) KIT	213
ตัวอย่างโปรแกรมอ่านและแสดงค่า	215
ตัวอย่างโปรแกรมดิจิทัลโพรบ	216
สรุป	217
บทที่ 18 การใช้งาน ET-ADC VR-NTC (NTC 10K) KIT	**219**
แนะนำบอร์ด ET-ADC VR-NTC (NTC 10K) KIT	219
เอ็นทีซี	220
พีทีซี	220
วงจรและการทำงานของบอร์ด ET-ADC VR-NTC (NTC 10K) KIT	221
ตัวอย่างโปรแกรมอ่านและแสดงค่า	224
สรุป	226
บทที่ 19 การใช้งาน ET-IR SENSOR KIT (BREAK BEAM)	**227**
แนะนำบอร์ด ET-IR SENSOR KIT (BREAK BEAM)	227
วงจรและการใช้งาน ET-IR SENSOR KIT (BREAK BEAM)	228
ตัวอย่างโปรแกรมเตือนเมื่อเกิดการตัดสัญญาณ	228
ตัวอย่างโปรแกรมนับค่า	230
สรุป	231
บทที่ 20 การใช้งาน ET-IR SENSOR KIT(reflec)	**233**
แนะนำบอร์ด ET-IR SENSOR KIT (reflec)	233
วงจรและการใช้งาน ET-IR SENSOR KIT (reflec)	233
ตัวอย่างโปรแกรมอ่านค่าความเข้มแสงที่สะท้อนกลับ	236
ตัวอย่างโปรแกรมตรวจสอบพื้นที่ขาวหรือดำ	237
สรุป	238
บทที่ 21 การใช้งาน ET-ADC SENSOR KIT (HALL EFFECT)	**239**
แนะนำบอร์ด ET-ADC SENSOR KIT (HALL EFFECT)	239
วงจรและการทำงานของบอร์ด ET-ADC SENSOR KIT (HALL EFFECT)	240
ตัวอย่างโปรแกรมอ่านและแสดงค่า	241

เนื้อหา	หน้า
ตัวอย่างโปรแกรมแสดงค่า Gauss	243
สรุป	244
บทที่ 22 การใช้งาน ET-ADC SENSOR KIT (TEMP)	**245**
แนะนำบอร์ด ET-ADC SENSOR KIT (TEMP)	245
วงจรและการทำงานของบอร์ด ET-ADC SENSOR KIT (TEMP)	246
ตัวอย่างโปรแกรมอ่านและแสดงค่าอุณหภูมิ	246
สรุป	248
บทที่ 23 การใช้งาน ET-ADC SENSOR KIT (LDR)	**249**
แนะนำบอร์ด ET-ADC SENSOR KIT (LDR)	249
วงจรและการทำงานของบอร์ด ET-ADC SENSOR KIT (LDR)	251
ตัวอย่างโปรแกรมอ่านและแสดงค่า	252
สรุป	253
บทที่ 24 การใช้งาน ET-DC MOTOR DRIVE	**255**
แนะนำบอร์ด ET-DC MOTOR DRIVE	255
วงจรและการทำงานของบอร์ด ET-DC MOTOR DRIVE	256
ตัวอย่างโปรแกรมสั่งงานมอเตอร์ไฟฟ้ากระแสตรง	258
ตัวอย่างโปรแกรมควบคุมการหมุนใบพัด	260
สรุป	262
บทที่ 25 การใช้งานบอร์ด LCD 16x2 I^2C	**263**
แนะนำบอร์ด LCD 16x2 I^2C	263
แอลซีดี	264
ส่วนควบคุมการทำงานแอลซีดี	265
การสั่งงานแบบ 4 บิต	268
ขั้นตอนการสั่งแอลซีดีตัวอักษร	272
วงจรบอร์ด LCD 16x2 I^2C	273
ตัวอย่างโปรแกรมแสดงผลข้อความ 2 บรรทัด	274
สรุป	277
บทที่ 26 การใช้งาน SOIL MOISTURE SENSOR	**279**
แนะนำบอร์ด SOIL MOISTURE SENSOR	279
ตัวอย่างโปรแกรมอ่านและแสดงค่า	281
สรุป	283

เนื้อหา	หน้า
บทที่ 27 การใช้งาน HC-SR501 BODY SENSOR	**285**
แนะนำบอร์ด HC-SR501 BODY SENSOR	285
ตัวอย่างโปรแกรมอ่านและแสดงค่า	288
สรุป	289
บทที่ 28 การใช้งาน MC-38 MAGNETIC SW	**291**
แนะนำ MC-38 MAGNETIC SW	291
ตัวอย่างโปรแกรมอ่านและแสดงค่า	292
สรุป	294
บทที่ 29 การใช้งาน MAX7219 Dot Matrix 8x32 Display	**295**
แนะนำบอร์ด MAX7219 Dot Matrix 8x32 Display	295
การเชื่อมต่อและหลักการทำงาน	296
ตัวอย่างโปรแกรมแสดงอักขระ	299
ตัวอย่างโปรแกรมสุ่มจุด	303
ตัวอย่างโปรแกรมเฟดอักษร	306
สรุป	310
บทที่ 30 การใช้งาน ET-MINI PCF8574	**311**
แนะนำบอร์ด ET-MINI PCF8574	311
วงจรของ ET-MINI PCF8574	313
ตัวอย่างโปรแกรมไฟวิ่งซ้าย/ขวา	314
ตัวอย่างโปรแกรมท็อกเกิล	317
ตัวอย่างโปรแกรมอ่านค่าจากสวิตซ์ 8 ตัว	319
ตัวอย่างโปรแกรมอ่านค่าจากสวิตซ์ 4x4	321
สรุป	326
บทที่ 31 การใช้งาน ET-MINI DS1307	**327**
แนะนำ ET-MINI DS1307	327
วงจรของ ET-MINI DS1307	329
การตั้งค่าเวลาและวันที่	329
การอ่านค่าเวลาและวันที่	331
ตัวอย่างโปรแกรมตั้งค่าเวลาและวันที่	331
ตัวอย่างโปรแกรมอ่านค่าเวลาและวันที่	332
สรุป	334

เนื้อหา	หน้า
บทที่ 32 การใช้งาน ET-BBC ROBOT และ ET-BBC ROBOT KIT	**335**
แนะนำ ET-BBC ROBOT และ ET-BBC ROBOT KIT	335
อุปกรณ์ต่าง ๆ	337
การควบคุม LED	339
การควบคุมเซอร์โวมอเตอร์	342
เดินหน้า	343
ถอยหลัง	344
เลี้ยวซ้าย	346
เลี้ยวขวา	348
หยุด	348
การตรวจจับเส้น	349
ถอยหลังเมื่อเข้าใกล้วัตถุ	353
ถอยเมื่อชนวัตถุ	355
สรุป	357
บทที่ 33 การสื่อสารผ่านสัญญาณวิทยุ	**359**
การสื่อสารผ่านสัญญาณวิทยุ	359
การสื่อสารทางเดียว	362
การสื่อสารสองทาง	364
การสั่งงานหุ่นยนต์แบบไร้สาย	365
สรุป	369
บรรณานุกรม	371

บทที่ 1

แนะนำ MicroPython

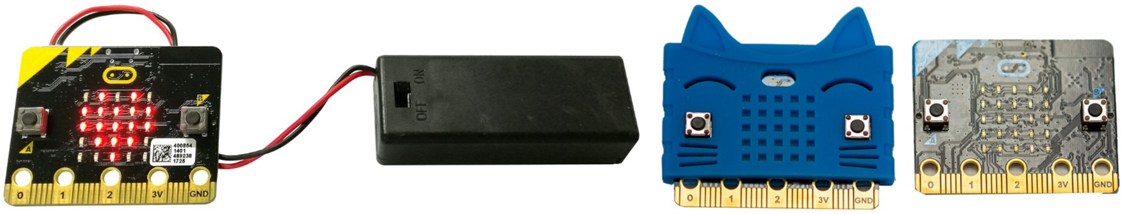
ในบทนี้แนะนำความเป็นมาและคุณสมบัติของบอร์ด BBC micro:bit บอร์ดขยายการทำงานของ micro:bit รุ่น ET-BBC GPIO V1 ซึ่งทำให้การใช้งานบอร์ด micro:bit มีความสะดวกยิ่งขึ้น พร้อมทั้งมีอุปกรณ์ต่อขยายสำหรับทดลองทั้งส่วนแสดงผลที่เป็น LED แบบ RGB สวิตซ์ ลำโพง ภาคจ่ายไฟและขาต่อขยาย นอกจากนี้ยังได้กล่าวถึงความเป็นมาของ MicroPython อันเป็นแกนของหนังสือเล่มนี้ เนื่องจากบอร์ด micro:bit ได้ติดตั้งเฟิร์มแวร์ MicroPython รุ่น 1.9x ที่ปรับปรุงเพื่อใช้กับ micro:bit เป็นการเฉพาะ ในตอนท้ายกล่าวถึงขั้นตอนการพัฒนาโปรแกรมสำหรับ micro:bit ด้วยภาษาไพธอนโดยใช้เครื่องมือ REPL ผ่าน rshell หรือชุดเขียนโปรแกรมอันได้แก่ Thonny และ Mu สำหรับเขียนโค้ดและอัพโหลดโปรแกรมจากเครื่องคอมพิวเตอร์ไปเก็บที่บอร์ด micro:bit ซึ่งเป็นเครื่องมือพื้นฐานสำหรับหนังสือเล่มนี้

แนะนำ BBC micro:bit

ไมโครบิต หรือ micro:bit เป็นบอร์ดไมโครคอนโทรลเลอร์ขนาดเล็ก (ภาพที่ 1-1) ที่ถูกออกแบบและพัฒนาขึ้นสำหรับการศึกษาเป็นการเฉพาะซึ่งถูกพัฒนาและสร้างโดยบริษัทแพร่ภาพและกระจายเสียงอังกฤษ หรือ BBC เพื่อแจกจ่ายแก่นักเรียนจำนวน 1 ล้านชุด ด้วยการร่วมมือกับหลายองค์กรได้แก่ บริษัทไมโครซอฟท์ (Microsoft) มหาวิทยาลัยแลนแคสเตอร์ บริษัทอีลีเม็น 14 (Element 14) และบริษัทซัมซุง (Samsung) จุดประสงค์เพื่อให้นักเรียนได้มีความคุ้นชินกับการเขียนโปรแกรมและอุปกรณ์อิเล็กทรอนิกส์ และเมื่อนักเรียนเหล่านี้เติบโตจะได้มีโอกาสทำงานและขับเคลื่อนธุรกิจด้านไอทีต่อไป

การเขียนโปรแกรมสำหรับบอร์ด micro:bit สามารถใช้ได้ 3 ภาษา คือ ภาษา JavaScript การเขียนผังงานแบบบล็อก (Block) และภาษา Python โดยทั้งสองมีข้อดีแตกต่างกันไป สำหรับการเขียนโปรแกรมด้วยภาษา JavaScript สามารถทำได้ผ่านเว็บบราวเซอร์และสามารถปรับเปลี่ยนการเขียนจากที่เป็นโค้ดให้เป็นแบบบล็อกได้

เพื่อสนับสนุนการเขียนโปรแกรมสำหรับมือใหม่หรือเด็กที่ยังไม่ค่อยเข้าใจการเขียนโปรแกรม ส่วนการเขียนโปรแกรมด้วยภาษา Python มีข้อดีคือสามารถปรับแต่งรายละเอียดได้ลึกและครอบคลุมมากกว่า เมื่อเขียนได้คล่องแล้วสามารถพัฒนาโปรแกรมได้รวดเร็วกว่า แต่ก็จำเป็นต้องใช้ความชำนาญที่มากกว่าด้วย หลังจากเขียนโปรแกรมเสร็จสามารถอัพโหลดโปรแกรมเข้าบอร์ด micro:bit ได้อย่างง่ายดายด้วยการเชื่อมต่อผ่านพอร์ตยูเอสบี (USB Port) โดยไม่ต้องซื้ออุปกรณ์เพิ่มเติม



ภาพที่ 1-1 บอร์ด BBC micro:bit

คุณสมบัติของ micro:bit

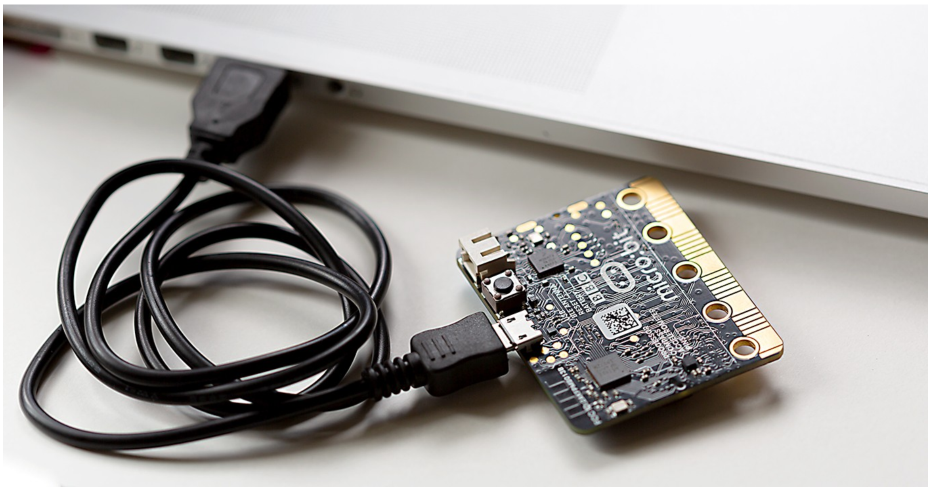
1. หน่วยไมโครคอนโทรลเลอร์หลักเบอร์ NRF51822 ของบริษัท NORDIC
   - ไมโครคอนโทรลเลอร์แบบ 32 บิต สถาปัตยกรรม ARM Cortex-M0
   - หน่วยความจำรอมประเภทแฟลชความจุ 256 KB
   - หน่วยความจำแรมความจุ 16 KB
   - ทำงานที่สัญญาณนาฬิกา 16 MHz
   - มีวงจร Bluetooth Low Energy (BLE)
2. หน่วยไมโครคอนโทรลเลอร์ที่สองมีหน้าที่เชื่อมพอร์ตยูเอสบี (USB) ซึ่งเป็นบอร์ด KL26Z ของบริษัท NXP/FREESCALE
   - ไมโครคอนโทรลเลอร์ขนาด 32 บิต สถาปัตยกรรม ARM Cortex-M0
   - ทำงานที่สัญญาณนาฬิกา 48 MHz
   - เชื่อมต่อพอร์ตยูเอสบีความเร็วตามมาตรฐาน USB2 (OTG)
3. ไอซีเบอร์ MMA8652 สำหรับเป็นเซ็นเซอร์ Accelerometer ทำหน้าที่วัดความเร่ง แบบ 3 แกน
4. ไอซีเบอร์ MAG3110 สำหรับเป็นเซ็นเซอร์ Magnetometer ทำหน้าที่วัดสนามแม่เหล็ก แบบ 3 แกน
5. หัวเชื่อมต่อพอร์ตยูเอสบีแบบ micro USB ใช้เป็นแหล่งจ่ายไฟเข้าบอร์ดและใช้ ดาวน์โหลดโปรแกรมจากคอมพิวเตอร์ลงบอร์ด micro:bit
6. มีขั้วต่อ POWER ประเภท JST แบบ 2 ขาจากแบตเตอรี่แรงดัน 3V

7. แอลอีดีสีแดงแบบเมตริกซ์จำนวน 5x5 หลอด สำหรับเป็นส่วนแสดงผลหรือดิสเพลย์ (DISPLAY) หลักของบอร์ด micro:bit
8. ขั้วต่อ I/O ภายนอก 23 ขา ซึ่งเป็นขาแบบ SLOT PCB ชุบทองสำหรับใช้ต่ออุปกรณ์ภายนอก
9. มีสวิตช์จำนวน 3 ตัว สำหรับเป็นสวิตช์ Reset และ button_a กับ button_b
10. ใช้ไฟ +5VDC จากขั้ว microUSB หรือ จากขั้ว 2 PIN JST
11. บอร์ด micro:bit มีขนาด 52 x 42 mm.

แต่อย่างไรก็ดี ด้วยประสิทธิภาพของหน่วยไมโครคอนโทรลเลอร์หลักที่ทำงานความถี่ 16MHz และมีปริมาณหน่วยความจำรอมและแรมที่ไม่มากนัก ผู้พัฒนา MicroPython สำหรับบอร์ด micro:bit จึงปรับปรุงโค้ดขึ้นใหม่เพื่อให้ใช้กับปริมาณหน่วยความจำที่น้อยกว่าบอร์ดที่รองรับ MicroPython อื่น ๆ และอาจส่งผลต่อการเขียนโปรแกรมที่ต้องใช้ความเร็วในการสื่อสาร เช่น บัสแบบ 1-Wire หรือการรับส่งข้อมูลกับรีโมทคอนโทรลแบบอินฟราเรด (IR Remote Control) เป็นต้น

แนะนำ ET-BBC GPIO V1

โดยปกติบอร์ด micro:bit เมื่อเสียบกับสาย microUSB และเชื่อมต่อกับเครื่อง PC ดังภาพที่ 1-2 ก็เพียงพอต่อการเขียนโปรแกรม แต่ในกรณีที่ต้องการเชื่อมต่อกับอุปกรณ์ภายนอกจะพบว่าพอร์ตบรบอร์ด micro:bit นั้นสามารถต่อเชื่อได้สะดวกเฉพาะ P0 P1 และ P2 แต่ขาอื่น ๆ มีขนาดเล็กจึงต้องอาศัยการต่อขยายบอร์ด

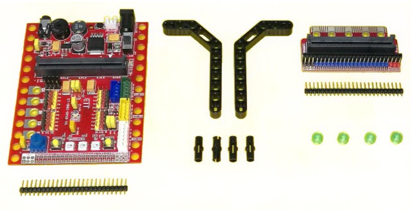


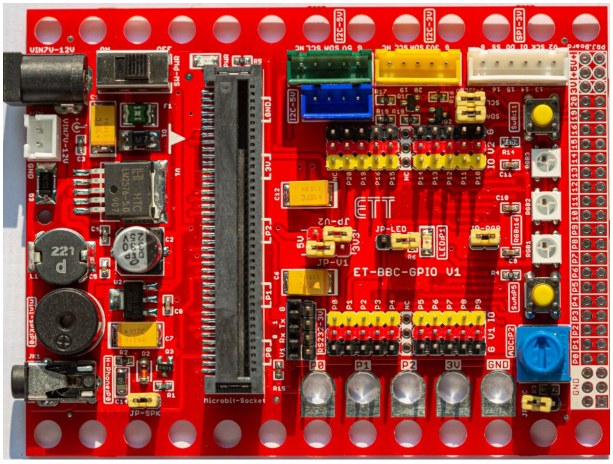
ภาพที่ 1-2 การเชื่อมต่อบอร์ด micro:bit กับเครื่อง PC

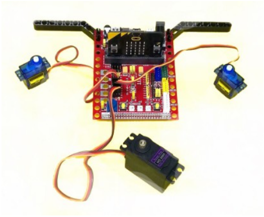
บอร์ด ET-BBC GPIO V1 (ภาพที่ 1-3) เป็นบอร์ดทีทาง อีทีที ผลิตขึ้นเพื่อแปลงขา Slot ของบอร์ด BBC micro:bit ให้สามารถนำไปต่อใช้งานได้โดยง่าย และสะดวกพร้อมมีวงจรพื้นฐานในการใช้งาน เหมาะสำหรับผู้ที่ใช้ในงานศึกษาบอร์ด BBC micro:bit ที่ควรจะมีใช้ โดยมีคุณสมบัติดังนี้

1. ซ็อกเก็ตใช้เสียบ SLOT ของบอร์ด BBC micro:bit ในแบบเสียบบอร์ดตั้งขึ้น
2. พร้อมชุด ET-BBC ADAPTER-V จะมีให้พร้อมในชุด ในกรณีลูกค้าจะเสียบบอร์ด BBC micro:bit ในแนวนอน
3. ขั้วต่อแบบ 3PIN แบบมาตรฐาน Servo JR โดยเป็นขั้ว GND, Vcc และ IO ต่อจากขั้ว P0 ถึง P20 ออกมายังขั้วต่อ 3 PIN พร้อมมีจัมเปอร์ (Jumper) สำหรับเลือกไฟ Vcc ให้ขา Vcc ของแถวใดเป็นแบบ 3.3V หรือ 5V
4. ขั้วต่อ I^2C ต่อกับอุปกรณ์ภายนอกมีให้แบบ 5PIN I^2C แบบ 3.3V พร้อมวงจรเปลี่ยนสัญญาณจาก I^2C 3.3V เป็น 5V แบบ 4PIN และแบบ 5PIN ทำให้ต่อกับอุปกรณ์ I^2C ที่เป็น 5V ได้โดยตรง
5. ขั้วต่อ SPI TTL-3V แบบขั้วต่อ 6 PIN
6. ขั้วต่อ RS232 TTL-3V (P1, P8) โดยเป็น PIN HEADER ที่มีให้ทั้งแบบตัวผู้ 4 PIN, และตัวเมีย 4 PIN
7. วงจรขับลำโพง พร้อมลำโพงขนาดเล็กและขั้วต่อ JACK หูฟัง ใช้ต่อหูฟังหรือลำโพง ได้โดยตรงเลย

8. ตัวต้านทานปรับค่าได้ หรือ VR ปรับค่า 10K Ohm บนบอร์ดจำนวน 1 ชุด สำหรับใช้ทดสอบการทำงานการแปลงสัญญาณแอนาล็อกเป็นดิจิทัล หรือ ADC (Analog to Digital Converter) ต่อเข้ากับ BBC micro:bit ที่ขา P2
9. มีแอลอีดี หรือ LED ชนิด RGB สามารถแสดงค่าสีได้ 16 ล้านสี ที่ควบคุมด้วยไอซีเบอร์ WS2812B จำนวน 3 หลอด สำหรับใช้ทดสอบการแสดงผล
10. มีสวิตช์ 2 ตัว ต่อจากขา P5 และ P11 ของ micro:bit เรียกว่า SW. A และ SW. B
11. ด้านข้างของบอร์ด ET-BBC GPIO V1 ถูกออกแบบให้มี ระยะรูและขนาดของรูตามแบบ LEGO ด้านละ 13 รู ทำให้ สามารถต่อเข้า กับชิ้น ส่วนต่าง ๆ ของ LEGO ในการยึดเข้ากับบอร์ด ET-BBC GPIO V1 ได้
12. มีรูสำหรับเสียบปากคีบ ตามมาตรฐานบอร์ด BBC micro:bit คือ P0, P1, P2, 3V และ GND
13. ด้านท้ายบอร์ดจะมีรู Pad Hole เป็น Pin I/O ทั้งหมดของ BBC micro:bit แบบ 2x25 Pin เพื่อนำไปบัดกรีต่อ Pin Header ตัวเมีย ใช้ต่อสาย หรืออาจบัดกรีต่อเป็น Pin Header แบบตัวผู้ใต้ PCB ใช้สำหรับต่อเข้ากับ PROJECT BOARD ได้
    - รู Pad Hole เป็น PIN IO ทั้งหมดของ BBC micro:bit แบบ 2x25 Pin คือ P0 ถึง P20, GND, VCC+3V และ VCC+5V
    - สามารถต่อเข้ากับ Project Board ได้โดยตรงต่ออุปกรณ์ทดลองต่าง ๆ ได้สะดวก
14. แจ็คดีซี (DC Jack) ขนาด 2.5 mm. และ Housing 2.5 mm. 2 PIN เป็นขั้วไฟจ่ายเข้าเลี้ยงบอร์ดเป็นไฟฟ้ากระแสตรงมีแรงดันในช่วง 7-12V พร้อมวงจรแปลงไฟให้ผลลัพธ์เป็น +5V ในแบบสวิตชิง (Switching Power Supply) ไม่ร้อนในการใช้งานและวงจรแปลงไฟ 3.3V นอกจากนี้ภาคจ่ายไฟยังมี Poly Switch เป็นฟิวส์ให้กับภาคจ่ายไฟอีกด้วย พร้อมสวิตช์สำหรับปิด/เปิดการทำงาน







ภาพที่ 1-3 บอร์ด ET-BBC GPIO V1

แนะนำ MicroPython

MicroPython เป็นโครงการที่นักฟิสิกส์ชาวออสเตรเลีย ชื่อ Damien P. George พัฒนาขึ้นเพื่อใช้ภาษาไพธอนเป็นภาษาสำหรับงานทางด้านไมโครคอนโทรลเลอร์ (microcontroller) ด้วยการสร้างโปรแกรมแปลภาษา Python 3 ให้เป็นรหัสเครื่องหรือเฟิร์มแวร์ (firmware) สำหรับไมโครคอนโทรลเลอร์โดยตรง และได้ดำเนินการระดมทุนเพื่อสร้างบอร์ด PyBoard ผ่านทางการระดมทุนของ Kickstarter (ปี ค.ศ. 2013-2014) และเผยแพร่โค้ดในรูปแบบเปิดเผยโค้ดบนข้อตกลงแบบ MIT (MIT License) หลังจากนั้น ปี ค.ศ. 2015 องค์กร ESA (European Space Agency) ของสหภาพยุโรปได้สนับสนุนโครงการ และทาง BBC (British Broadcasting Corporation) ได้นำ MicroPython ไปใช้กับบอร์ด micro:bit นอกจากนี้ Limor Ladylada (ผู้ก่อตั้ง Adafruit Industries) ของสหรัฐอเมริกาได้พัฒนาไลบรารี่สนับสนุน MicroPython สำหรับฮาร์ดแวร์ของทางบริษัท พร้อมทั้งพัฒนาบอร์ด Circuit Playground Express และพัฒนาต่อยอด MicroPython เป็น CircuitPython นอกจากนี้ ด้วยความสำเร็จของบอร์ดตระกูล ESP8266 และ ESP32 ของบริษัท Expressif Systems จึงมีการพัฒนาเฟิร์มแวร์ ให้สามารถใช้งาน MicroPython ได้

คุณสมบัติของ MicroPython ที่แตกต่างจาก Python 3 คือ โปรแกรมที่เขียนสามารถรัน (run) โดยตรงจากไมโครคอนโทรลเลอร์ พร้อมทั้งเข้าถึงฮาร์ดแวร์ เช่น GPIO SPI I^2C WiFi ได้ แต่ด้วยข้อจำกัดทางด้านปริมาณหน่วยความจำของไมโครคอนโทรลเลอร์จึงรองรับไลบรารี่ที่มีความจำเป็นเท่านั้น และรองรับ

การสื่อสารกับเฟิร์มแวร์ MicroPython ในแบบ Interactive ที่เรียกว่า REPL (Read, Event, Print, Loop) ผ่านทางพอร์ตสื่อสารแบบอนุกรม (ใช้ตัวแปลง USB-to-RS232)

ขั้นตอนการพัฒนาโปรแกรม

การพัฒนาโปรแกรมกับบอร์ด micro:bit ต้องอาศัยการเขียนโปรแกรมจากเครื่อง PC ซึ่งในหนังสือเล่มนี้เลือกใช้โปรแกรม Thonny และ MU เป็นเครื่องมือในการทดสอบโปรแกรมของหนังสือ โดยวิธีการใช้งานจะกล่าวละเอียดในหัวข้อของการใช้ Thonny และ MU

ขั้นตอนในการพัฒนาโปรแกรมเป็นดังนี้

1. เชื่อมต่อบอร์ด micro:bit กับเครื่อง PC
2. เรียกใช้โปรแกรมสำหรับเขียนโปรแกรม เช่น Thonny หรือ Mu
3. เขียนโค้ดในโปรแกรม
4. สั่งรันโปรแกรม ถ้าพบข้อผิดพลาดให้กลับไปแก้ไขโปรแกรม
5. ดูผลการทำงานว่ามีความถูกต้องตามความต้องการหรือไม่ ถ้ายังไม่ครบถ้วนให้กลับไปแก้ไขโปรแกรม หรือเขียนโค้ดเพิ่มเติม
6. การบันทึกโปรแกรมลงบอร์ด micro:bit กระทำโดยการสั่งบันทึกลงบอร์ด micro:bit (โปรแกรม Thonny) หรือกดไอคอน Flash (โปรแกรม Mu) และกรณีที่เป็นโปรแกรมหลักให้บันทึกไฟล์ชื่อ main.py

การใช้ Thonny

โปรแกรม Thonny เป็นซอฟต์แวร์ที่พัฒนาภายใต้หน่วยงาน Institute of Computer Science แห่งมหาวิทยาลัย Tartu ประเทศ Estonia เป็นโปรแกรม GUI (Graphics User Interface) ที่เน้นเรียบง่าย ทำหน้าที่เป็นโปรแกรม IDE ภาษาไพธอนที่ทำงานบนหลายระบบปฏิบัติการ และรองรับการทำงานในหลายโหมด ได้แก่ โหมดภาษาไพธอนรุ่น 3.7.7 โหมดการทำงานกับ MicroPython ของบอร์ด ESP8266/ESP32/BBC micro:bit/CircuitPython และโหมดการทำงานกับ MicroPython บนบอร์ดอื่น ๆ (generic) โดยรุ่นที่ใช้ในการเขียนเป็นรุ่น 3.2.7 (17 เมษายน พ.ศ. 2563)

การติดตั้งโปรแกรม Thonny

การติดตั้งโปรแกรมสามารถเลือกติดตั้งได้ 2 ลักษณะ คือ ดาวน์โหลดโปรแกรมใช้งานจากเว็บไซต์ www.thonny.org และเลือกดาวน์โหลดตามระบบปฏิบัติการที่ใช้งานซึ่งรองรับระบบปฏิบัติการวินโดวส์ แม็คโอเอส และลินุกซ์ สำหรับการติดตั้งในอีกรูปแบบหนึ่งคือใช้คำสั่ง pip ของไพธอน ดังนี้

```
pip install thonny
```

การเรียกใช้ Thonny จากการติดตั้งด้วย pip กระทำโดยการสั่งงานดังนี้

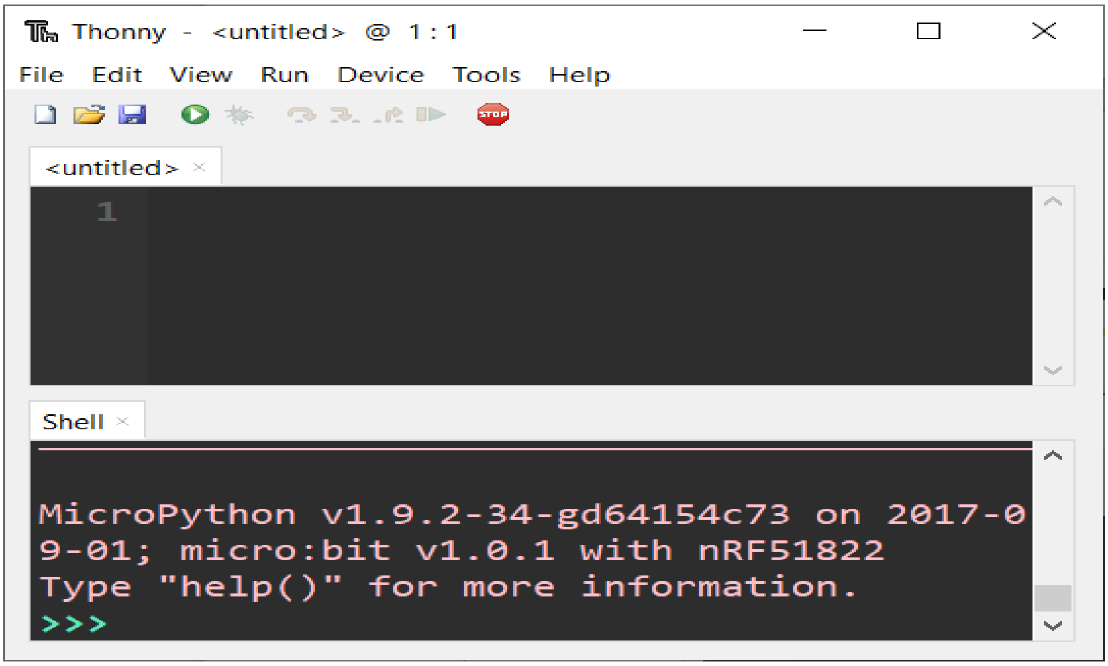
```
python -m thonny
```

การอัพเกรดโปรแกรม Thonny ในกรณีที่ติดตั้งด้วย pip สามารถสั่งอัพเกรดโดยคำสั่งดังต่อไปนี้

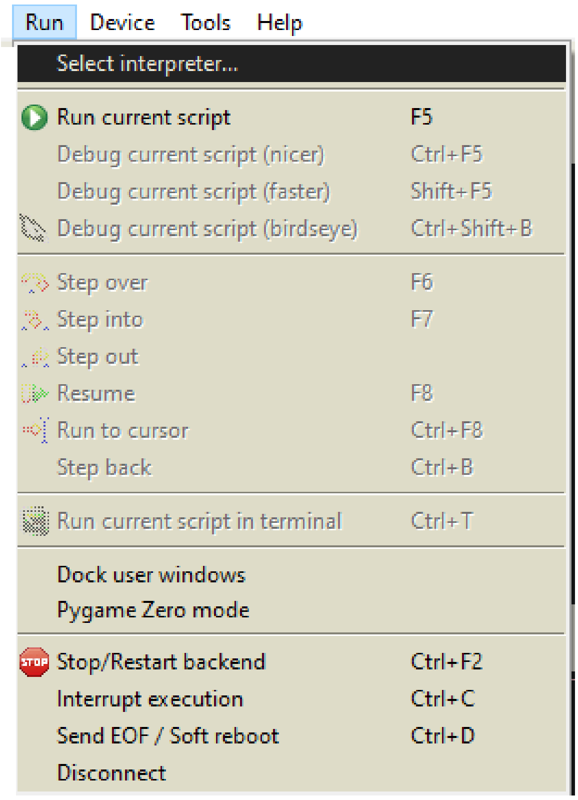
```
pip install --upgrade thonny
```

หน้าจอเมื่อเข้าโปรแกรม Thonny เป็นดังภาพที่ 1-4 โดยหน้าจอถูกแบ่งเป็น 2 ส่วน ด้านบนเป็นส่วนของการเขียนโค้ด และด้านล่างเป็น Shell หรือ REPL ทำให้สามารถสั่งงานได้ 2 แบบ คือ เขียนโค้ดแล้วส่งโค้ดไปรัน หรือเขียนโค้ดจาก shell แบบ REPL เหมือนใช้ rshell

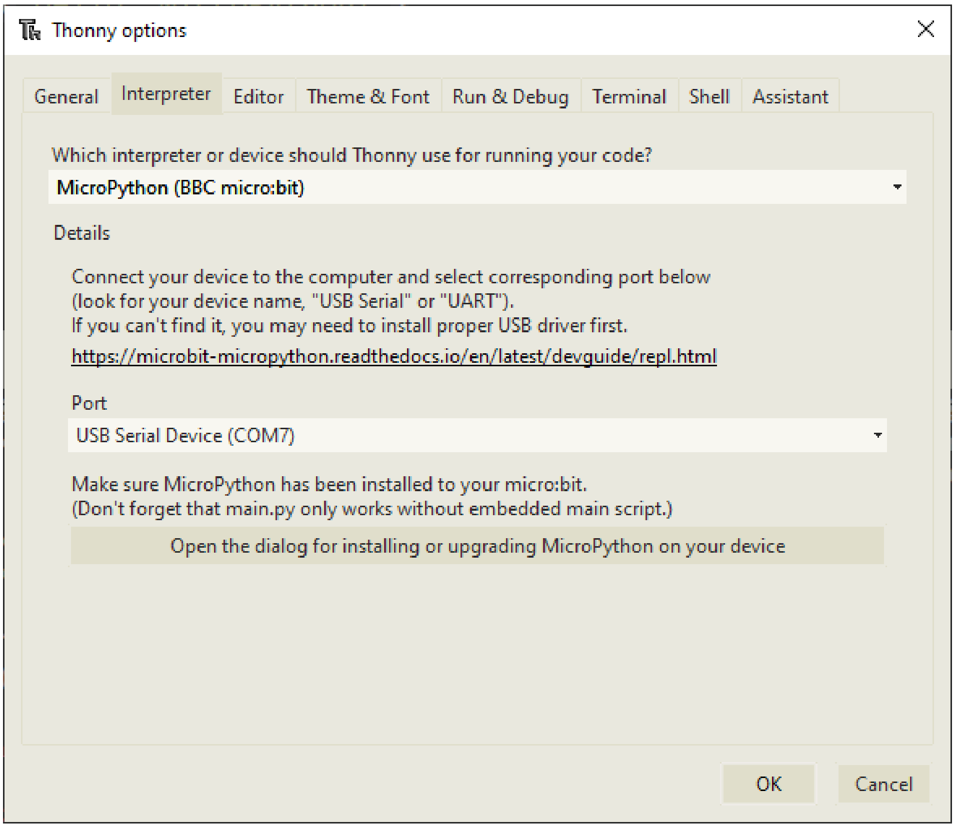
เนื่องจากโปรแกรม Thonny รองรับการทำงานในหลายโหมด ดังนั้น การเลือกโหมดเป็นขั้นตอนแรกที่ต้องตรวจสอบให้แน่ใจว่ามีความถูกต้อง โดยการเลือกนั้นทำโดยการเข้าเมนู Run และเลือกรายการย่อย Select interpreter... ดังภาพที่ 1-5 หลังจากนั้นหน้า Thonny options ปรากฏดังภาพที่ 1-6 ให้กำหดโหมดเป็น MicroPython (BBC micro:bit) และเลือกพอร์ตสื่อสารให้ถูกต้อง โปรแกรมก็พร้อมใช้งาน



ภาพที่ 1-4 หน้าจอโปรแกรม Thonny



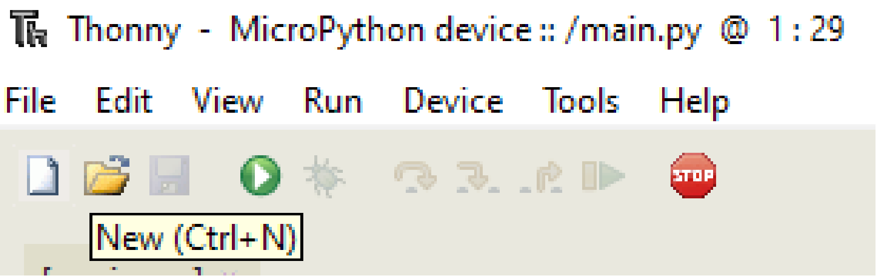
ภาพที่ 1- หน้าจอเมนู Run/Select Interpreter... ของโปรแกรม Thonny



ภาพที่ 1-6 หน้าจอ Thonny options

ไอคอนสั่งงานของ Thonny

โปรแกรม Thonny มีไอคอนสั่งงานโปรแกรมด้วยกัน 10 รายการ แต่สามารถใช้กับ MicroPython ได้ 4 รายการ คือ สร้างไฟล์โปรแกรมใหม่ (New) เปิดไฟล์โปรแกรม (Open) อัพโหลดโปรแกรมแล้วรัน (Run) และหยุดการทำงานของ MicroPython และกลับสู่การรอรับคำสั่ง (Stop) ดังภาพที่ 1-7

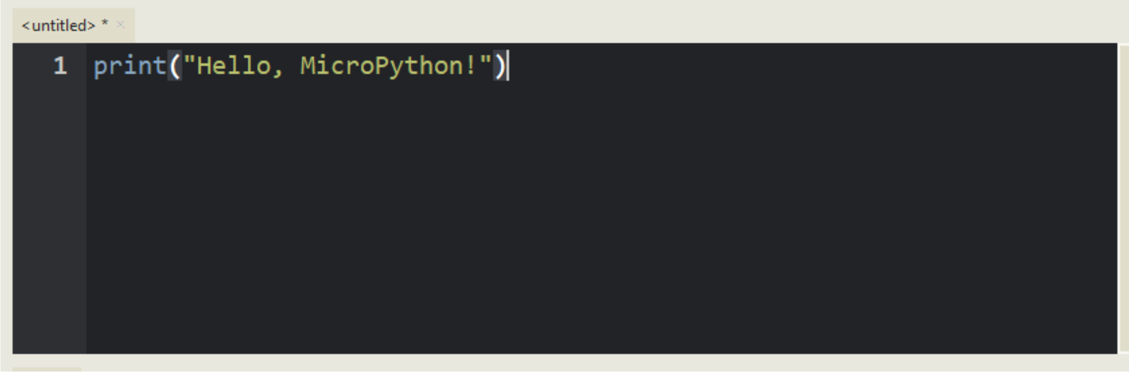


ภาพที่ 1-7 ไอคอนคำสั่งของโปรแกรม Thonny

การเขียนโค้ดและรันโปรแกรมของ Thonny

การเขียนโค้ดเริ่มต้นด้วยการสั่ง New ด้วยการเลือกจากรายการเมนู File และเลือกรายการย่อย New หรือคลิกที่ไอคอนคำสั่ง New ได้เช่นกัน หลังจากนั้นจะได้แท็บของไฟล์ใหม่ชื่อว่า <untitled> เมื่อพิมพ์คำสั่งดังภาพที่ 1-8 จะมีเครื่องหมาย * ปรากฏหลังชื่อไฟล์ <untitled> โค้ดที่พิมพ์เพื่อทดสอบการทำงานเป็นคำสั่งแสดงข้อความ Hello, MicroPython! ผ่านทางการสื่อสารอนุกรม ซึ่งจะแสดงผลใน shell ของ REPL ชุดคำสั่งที่พิมพ์ คือ

```
print("Hello, MicroPython!")
```

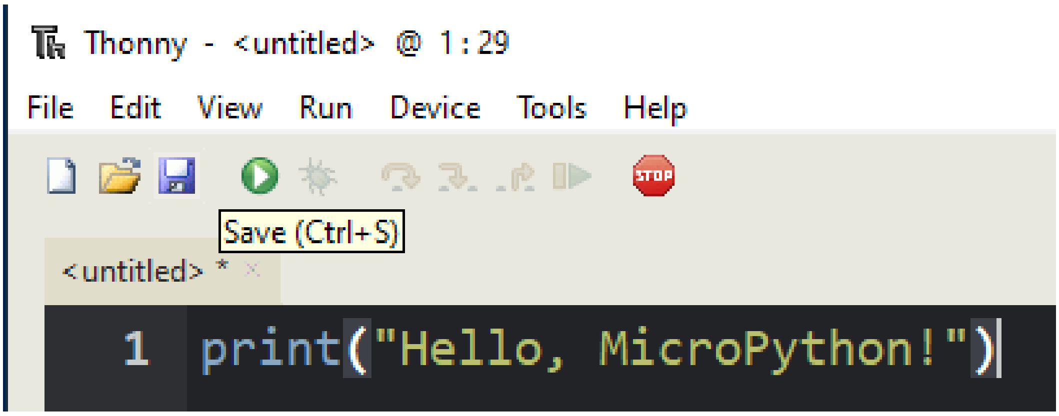


ภาพที่ 1-8 หน้าจอเขียนโค้ดของ Thonny

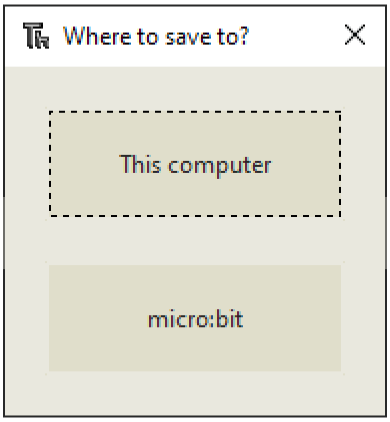
เมื่อได้โค้ดโปรแกรมให้ทำการบันทึกไฟล์โปรแกรม ด้วยการคลิกที่ปุ่ม Save ดังภาพที่ 1-9 หรือเลือกรายการคำสั่งบันทึกจากเมนู File ซึ่งมีให้เลือก 3 รูปแบบ คือ

Save	บันทึกลงชื่อไฟล์เดิม
Save as..	บันทึกลงชื่อไฟล์ใหม่ และเปิดไฟล์ใหม่ (ปิดไฟล์เดิม)
Save copy	สำเนาลงไฟล์ใหม่ โดยเปิดไฟล์เดิม

เนื่องจากการเขียนโปรแกรมเป็นการเขียนโปรแกรมเพื่อสั่งงานให้บอร์ด micro:bit ซึ่งถือว่าเป็นอุปกรณ์ที่ติดตั้งเฟิร์มแวร์ MicroPython ทำงาน การบันทึกไฟล์โปรแกรมจึงต้องเลือกระหว่างบันทึกลงเครื่องคอมพิวเตอร์หรือบันทึกลงบอร์ด micro:bit ดังภาพที่ 1-10 ซึ่งในขั้นตอนการพัฒนาสามารถบันทึกเก็บไว้ในเครื่องคอมพิวเตอร์แต่การติดตั้งระบบงานจริงต้องบันทึกไฟล์ลงในรอมของอุปกรณ์



ภาพที่ 1-9 ไอคอนบันทึกไฟล์ของโปรแกรม Thonny



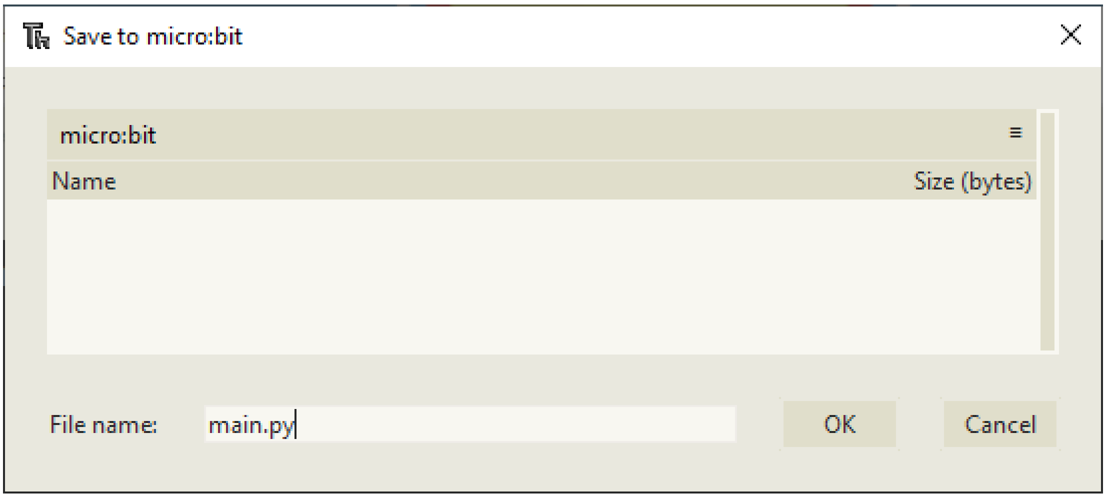
ภาพที่ 1-10 หน้าจอเลือกแหล่งจัดเก็บโปรแกรม micro:bit ของ Thonny

หมายเหตุ

การบันทึกไฟล์โปรแกรมลงบอร์ด micro:bit ในบางครั้งการบันทึกอาจไม่แสดงหน้าจอตามภาพที่ 1-10 เนื่องจากบอร์ด micro:bit อยู่ในสถานะการทำงานที่วนรอบไม่รู้จบหรือกำลังประมวลผลตามคำสั่งที่เขียนไว้ทำให้ไม่สามารถตอบสนองกับโปรแกรม Thonny ได้ตามปกติผู้อ่านต้องคลิกที่ไอคอน Stop เพื่อหยุดการทำงาน แต่ถ้าไม่สำเร็จจะต้องถอดสาย USB แล้วเสียบใหม่

กรณีที่เลือกบันทึกลงเครื่อง PC จะแสดงหน้าจอสำหรับบันทึกไฟล์ปกติ แต่ถ้าเลือกเป็นบันทึกลงอุปกรณ์จะปรากฏหน้าต่างดังภาพที่ 1-11 ซึ่งรายงานชื่อไฟล์ที่บันทึกอยู่ในรอมของบอร์ด micro:bit ดังในภาพตัวอย่างไม่มีไฟล์ถูกบันทึกลงบอร์ด micro:bit มาก่อนจึงไม่แสดงรายชื่อไฟล์ในหน้าต่าง และในขั้นตอนนี้ให้ใส่ชื่อไฟล์ลงในช่องหลัง

ข้อความ File name แล้วคลิก OK เพื่อดำเนินการบันทึก หรือคลิก Cancel เพื่อยกเลิกการสั่งงานบันทึกไฟล์



ภาพที่ 1-11 หน้าจอตั้งชื่อไฟล์บน micro:bit ของโปรแกรม Thonny

หมายเหตุ

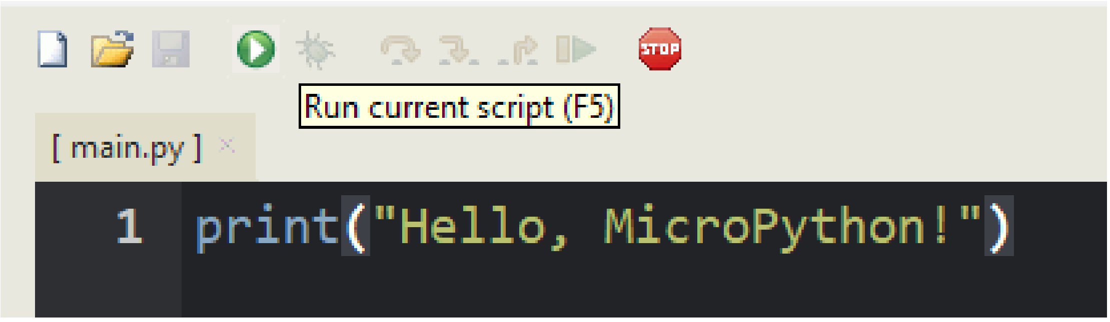
1.ไฟล์โปรแกรมชื่อ main.py เป็นชื่อไฟล์ที่ตัวแปลภาษาไพธอนกำหนดไว้ให้เป็นชื่อของโปรแกรมที่ถูกเรียกทำงานเมื่อระบบเริ่มต้นทำงาน ดังนั้น ต้องตั้งชื่อโปรแกรมหลักบนบอร์ด micro:bit เป็นชื่อ main.py เสมอ

2.ในหน้าต่างภาพที่ 1-11 ผู้อ่านสามารถคลิกขวาเพื่อให้ขึ้นเมนูสำหรับสั่งลบไฟล์ ลบไดเร็กทอรี หรือสร้างไดเร็กทอรีได้

หลังจากบันทึกไฟล์โปรแกรมสำเร็จให้สังเกตที่ชื่อไฟล์ของส่วนเขียนโปรแกรมปรากฏเครื่องหมายวงเล็บ [และ] ด้านหน้า และด้านหลังชื่อไฟล์เพื่อเป็นการบ่งบอกให้ทราบว่าเป็นไฟล์ที่อยู่ในบอร์ด micro:bit ไม่ใช่โค้ดที่บันทึกลงเครื่อง PC ดังภาพที่ 1-12

การรันโปรแกรมกระทำได้ 2 ลักษณะ คือ กรณีที่เป็นโปรแกรมที่จัดเก็บบนเครื่อง PC ตัวโปรแกรม Thonny จะขอจองบัฟเฟอร์จากบอร์ด micro:bit และส่งโค้ดทั้งหมดไปเพื่อรันโปรแกรม และกรณีที่โค้ดอยู่ในบอร์ด micro:bit อยู่แล้วจะเป็นการสั่งรันจากบอร์ดได้ทันที ซึ่งกรณีที่โค้ดมีขนาดใหญ่จนเกินปริมาณที่โปรแกรม Thonny ขอจองจากบอร์ด micro:bit ได้ จะเกิดปัญหาไม่สามารถรันโปรแกรมได้เนื่องจากไม่มีพื้นที่เพียงพอให้ส่งโปรแกรมไปจัดเก็บ ดังนั้น การรันโปรแกรมที่ถูกบันทึกลงบอร์ด micro:bit จะแก้ปัญหาเรื่องของการจองบัฟเฟอร์ เมื่อโปรแกรมเริ่มทำงานจากการคลิกที่ไอคอน Run current script หรือเลือกจากเมนู Run และเลือกเมนูย่อย Run current script

หรือกดแป้นลัด F5 จะเกิดผลลัพธ์ตามภาพที่ 1-13 แต่ถ้าบันทึกไฟล์เป็น main.py และบันทึกลงอุปกรณ์ MicroPython เมื่อผู้ใช้กดปุ่ม reset ของบอร์ด จะทำให้ระบบถูกเริ่มต้นใหม่ และโค้ดของโปรแกรม main.py จะถูกเรียกให้ทำงาน จึงเกิดผลลัพธ์เหมือนภาพที่ 1-14 สำหรับในกรณีที่หน้าต่าง shell หายไปจากหน้าต่างโปรแกรม Thonny ผู้ใช้สามารถเรียกกลับหน้าต่าง shell ให้ปรากฏอีกครั้งด้วยการไปที่เมนู View และสังเกตเครื่องหมาย √ หน้ารายการ Shell ดังภาพที่ 1-15 ถ้าเครื่องหมายหายไปให้คลิกที่เมนู Shell



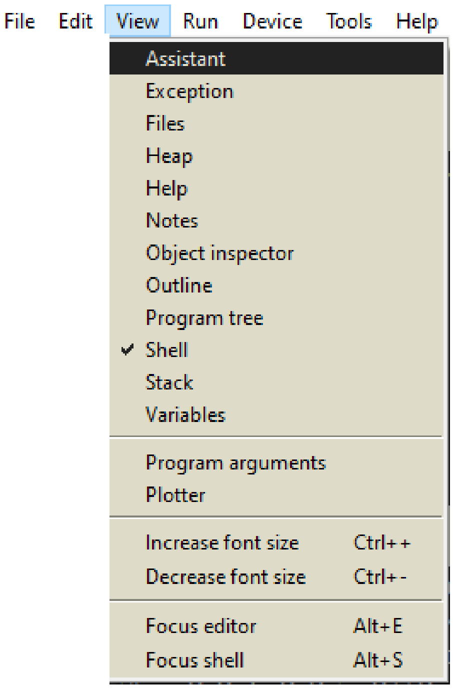
ภาพที่ 1-12 หน้าจอแสดงชื่อไฟล์ที่ถูกบันทึกลงบอร์ด micro:bit ของโปรแกรม Thonny

```
>>> %Run -c $EDITOR_CONTENT
 Hello, MicroPython!
>>>
```

ภาพที่ 1-13 หน้าจอการรันโปรแกรมจากการคลิกไอคอน Run ของ Thonny

```
>>> Hello, MicroPython!
 MicroPython v1.9.2-34-gd64154c73 on 2017-09-01;
 micro:bit v1.0.1 with nRF51822
 Type "help()" for more information.
>>>
```

ภาพที่ 1-14 ผลของการรันโปรแกรม main.py เมื่อกดปุ่ม reset บนหน้าจอ Shell ของ Thonny



ภาพที่ 1-15 เมนูสั่งแสดง/ซ่อนหน้าต่างต่าง ๆ ของโปรแกรม Thonny

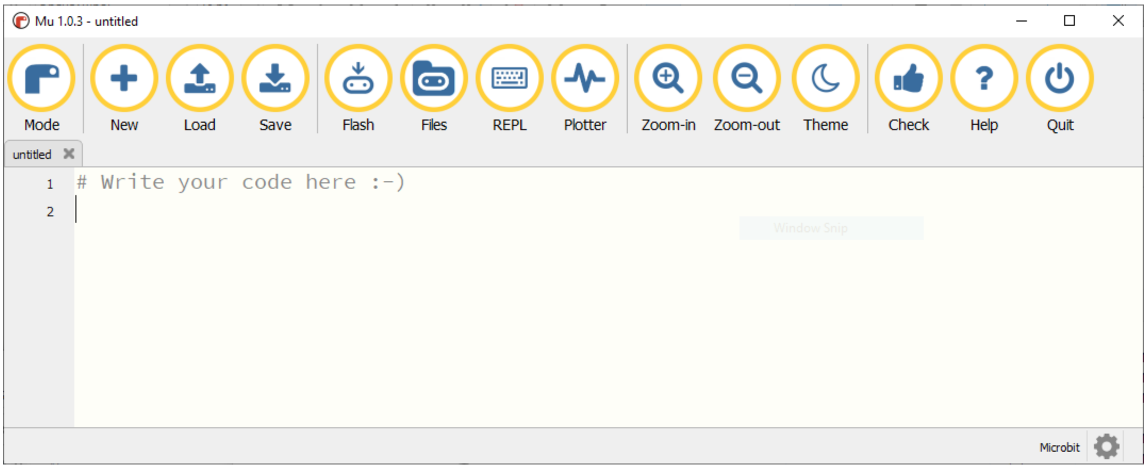
การใช้ Mu Editor

Mu เป็นโปรแกรมที่ทำหน้าที่เหมือนกับ Thonny แต่มีขั้นตอนการใช้งานที่ซับซ้อนน้อยกว่าและมีการประสานการใช้งานกับผู้ใช้แตกต่างกัน ดังภาพที่ 1-16 รองรับการทำงานกับบอร์ดที่เป็นอุปกรณ์ของ MicroPython ดังภาพที่ 1-17 อันได้แก่ Adafruit CircuitPython, BBC micro:bit, Pygame Zero และ Python3 ซึ่งเครื่องมือ Mu ออกแบบเพื่อใช้สำหรับการเรียนการสอนแบบมีการตอบสนองระหว่างกันระหว่างผู้เรียนและผู้สอน

ไอคอนสั่งงานของโปรแกรม Mu จากภาพที่ 1-17 ประกอบด้วย

- Mode เป็นคำสั่งเลือกโหมดการทำงานของโปรแกรม Mu
- New สำหรับสร้างไฟล์โปรแกรมใหม่
- Load สำหรับเปิดไฟล์ที่เขียนไว้ก่อนหน้านี้เพื่อกลับมาทำงานต่อ
- Save สำหรับบันทึกไฟล์ลงเครื่อง PC
- Zoom In สำหรับขยายขนาดตัวอักษร
- Zoom Out สำหรับลดขนาดของตัวอักษร
- Theme สำหรับเลือกชุดสีของโปรแกรม

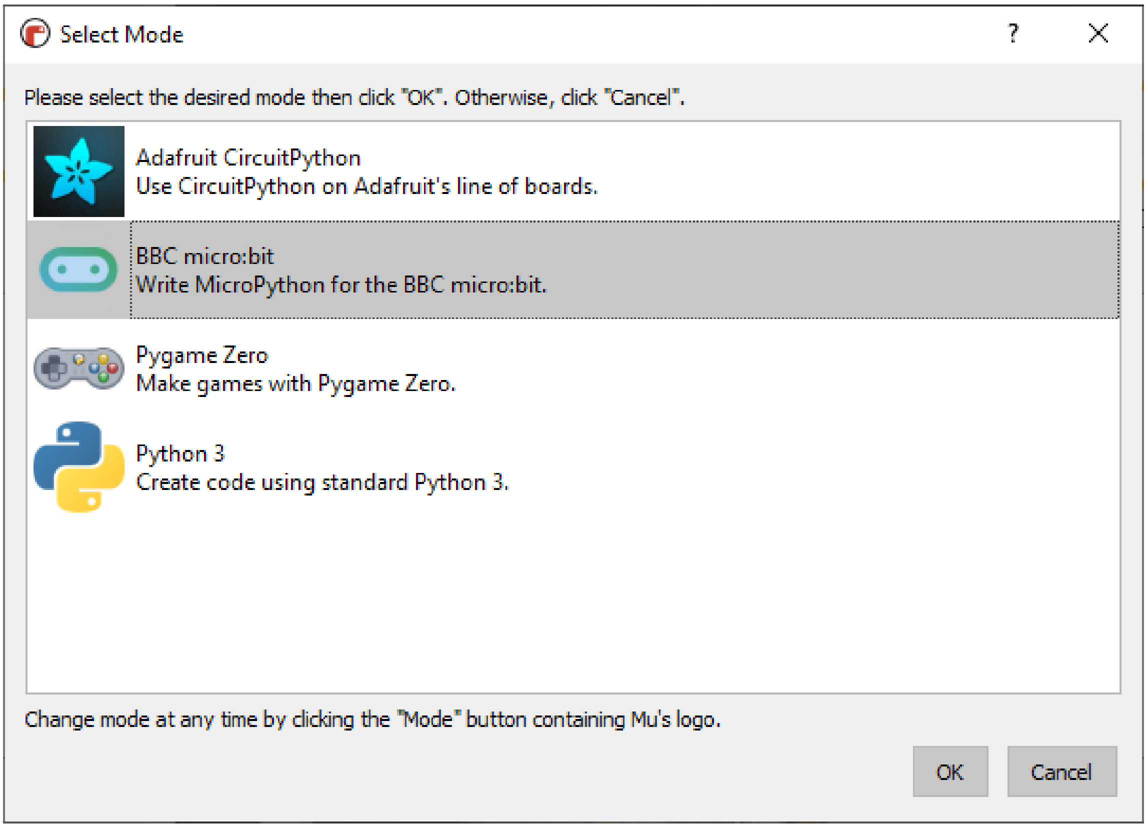
- Check สำหรับตรวจสอบโค้ดโปรแกรมที่กำลังเขียน
- Help สำหรับไปยังเว็บไซต์เพื่อขอข้อมูลช่วยเหลือ
- Quit สำหรับออกจากโปรแกรม



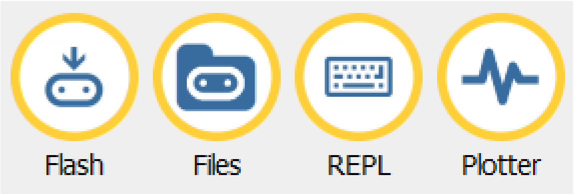
ภาพที่ 1-16 หน้าจอหลักของโปรแกรม Mu

เครื่องมือหลักของโปรแกรม Mu เมื่อใช้กับบอร์ด micro:bit เป็นดังภาพที่ 1-18 อันได้แก่เครื่องมือ Flash Files REPL และ Plotter โดยแต่ละเครื่องมือมีหน้าที่ดังนี้

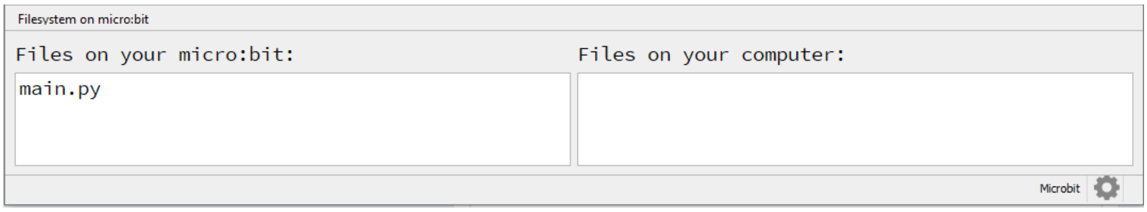
- Flash ใช้สำหรับโอนไฟล์โปรแกรมที่กำลังเขียนอยู่ในปัจจุบันส่งไปยังรอมของบอร์ด micro:bit พร้อมทำการเปลี่ยนชื่อไฟล์เป็น main.py แต่การจะ Flash ได้จะต้องไม่เปิดหน้าต่าง REPL และ Plotter
- Files ใช้สำหรับการโอนข้อมูลระหว่างบอร์ด micro:bit กับไดเร็กทอรีในเครื่อง PC ดังภาพที่ 1-19 โดยตำแหน่งของไฟล์ในเครื่อง PC อยู่ที่ C:/Users/ชื่อผู้ใช้/mu_code ซึ่งวิธีการโอนไฟล์กระทำโดยการลากและวางจากหน้าต่างหนึ่งไปยังอีกหน้าต่างหนึ่ง ส่วนการลบไฟล์ในบอร์ด micro:bit กระทำโดยคลิกขวาที่ชื่อไฟล์และเลือกเมนู Delete ดังภาพที่ 1-20
- REPL เป็นการเปิดหน้าต่างเพื่อใช้งานโปรแกรม MicroPython บนบอร์ด micro:bit หรือใช้สำหรับดูผลการทำงานจากคำสั่ง print ดังภาพที่ 1-21
- Plotter เป็นหน้าจอแสดงผลโดยใช้ข้อมูลจากคำสั่ง print แต่ต้องกำหนดข้อมูลเป็นพิกัด 2 มิติ เช่น print((adc_value,)) เป็นต้น โดยตัวอย่างผลลัพธ์ที่ได้จากหน้าต่าง Plotter เป็นดังภาพที่ 1-22



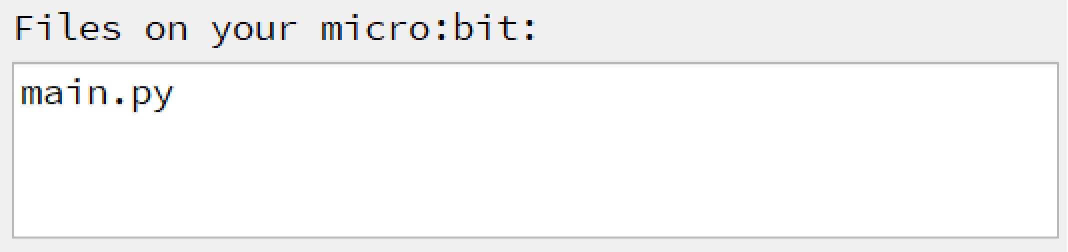
ภาพที่ 1-17 โหมดการทำงานของโปรแกรม Mu



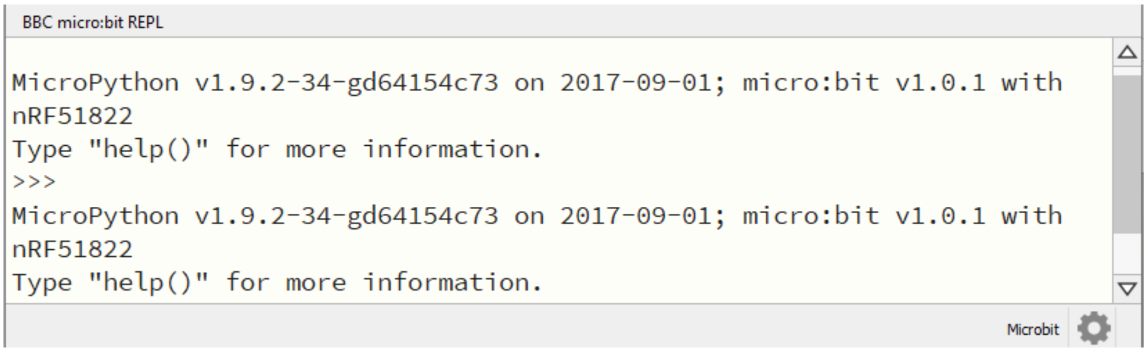
ภาพที่ 1-18 เครื่องมือสำหรับ micro:bit ของโปรแกรม Mu



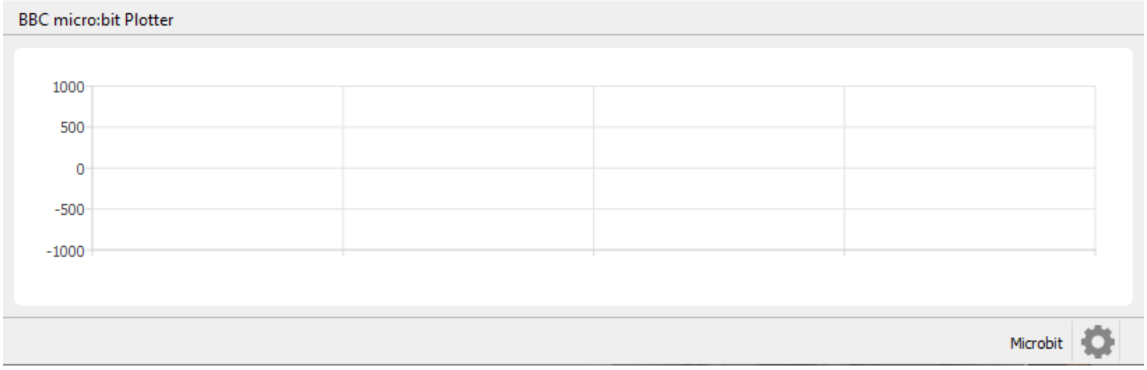
ภาพที่ 1-19 หน้าต่าง Files ของโปรแกรม Mu



ภาพที่ 1-20 เมนูสั่งลบไฟล์ของโปรแกรม Mu



ภาพที่ 1-21 หน้าจอ REPL ของโปรแกรม Mu



ภาพที่ 1-22 หน้าตาส่วนของการแสดงกราฟของโปรแกรม Mu

สรุป

จากบทนี้จะพบว่าบอร์ด BBC micro:bit เป็นบอร์ดขนาดเล็กที่ถูกออกแบบเพื่อการศึกษา มีประสิทธิภาพการทำงานที่ครอบคลุมทั้งการแสดงผล การรับค่าในแบบสัญญาณดิจิทัลและแอนาล็อก การนำออกสัญญาณแบบดิจิทัลและแอนาล็อก โดยมีความร่วมมือกับหลายยักษ์ใหญ่ทำให้มีเครื่องมือในการพัฒนาที่หลากหลาย เช่น บริษัทไมโครซอฟท์ที่ออกแบบโปรแกรมภาษาบล็อกทำให้ใช้งานได้ง่าย หรือทีมผู้แปลง MicroPython

ให้รองรับการทำงานของ micro:bit ทำให้สามารถเขียนโปรแกรมด้วยภาษาที่ไม่ซับซ้อนอย่างภาษาไพธอนได้ เป็นต้น

แต่อย่างไรก็ดี ด้วยขนาดของบอร์ดที่มีขนาดเล็ก และออกแบบเพื่อใช้งานขา pin0 pin1 และ pin2 เป็นส่วนใหญ่ ทำให้การใช้งานขาอื่น ๆ ของบอร์ด micro:bit จึงต้องมีการต่อวงจรเพิ่ม โดยในหนังสือเล่มนี้เลือกใช้บอร์ด ET-BBC GPIO V1 เป็นบอร์ดขยายขาเชื่อมต่อกับอุปกรณ์ภายนอกของบอร์ด micro:bit โดยตัวบอร์ดขยายนอกจากทำให้ต่อวงจรภายนอกได้ง่าย ยังมีอุปกรณ์ต่อพ่วงภายนอกติดตั้งไว้บนบอร์ด เช่น ลำโพงขนาดเล็ก หลอดแอลอีดีแบบอาร์จีบี และสวิตช์ เป็นต้น ทำให้สามารถทดลองเขียนโปรแกรมได้มากกว่าการใช้เพียงส่วนเชื่อมต่อบนบอร์ด micro:bit และในบอร์ดเสริมมีภาคจ่ายไฟทำให้สามารถต่อตัวแปลงไฟภายนอกมาใช้กับวงจรที่เชื่อมต่อเพื่อแก้ปัญหาแรงดันหรือกระแสไม่เพียงพอในการขับอุปกรณ์

การเขียนโปรแกรมภาษาไพธอนสำหรับบอร์ด micro:bit สามารถเขียนได้หลายเครื่องมือ ไม่ว่าจะเป็น rshell ที่เขียนผ่าน console โหมดของระบบปฏิบัติการ โปรแกรม Thonny และ Mu ที่ทำงานเป็นโหมดกราฟิกส์ที่รวบรวมเครื่องมือการสั่งงานไว้ในหน้าต่างโปรแกรมเดียวจึงสะดวกกว่าการสั่งงานด้วยมือทั้งหมด

แต่อย่างไรก็ดีการเขียนโปรแกรมจะต้องเกิดจากการคิด ออกแบบ และนำมาเขียนโค้ด ดังนั้น ในบทถัดไปเป็นเนื้อหาเกี่ยวกับหลักการเขียนโปรแกรมภาษาไพธอนในระดับพื้นฐานสำหรับการเขียนโปรแกรมสั่งงานบอร์ด micro:bit ให้สามารถทำงานได้หลากหลาย และสามารถนำโค้ดกลับมาใช้ได้โดยไม่ต้องเขียนใหม่ทั้งหมด นอกจากนี้ ในหนังสือยังมีเนื้อหาเกี่ยวข้องกับอุปกรณ์ภายนอกทั้งส่วนนำเข้าข้อมูลแบบต่าง ๆ เช่น สวิตช์ หรือเซ็นเซอร์อุณหภูมิในแบบต่าง ๆ เป็นต้น หรืออุปกรณ์แสดงผล เช่น แอลอีดีชนิดต่าง ๆ รีเลย์ เมตริกซ์แอลอีดี หรือแอลซีดีตัวอักษร เป็นต้น จึงหวังว่าหนังสือเล่มนี้จะเป็นจุดเริ่มต้นที่ดีสำหรับผู้ที่สนใจเขียนโปรแกรมภาษาไพธอนบนอุปกรณ์ micro:bit ต่อไปในอนาคต

บทที่ 2

ภาษาไพธอน

ในบทนี้เป็นการเรียนรู้พื้นฐานการเขียนโปรแกรมภาษาไพธอน ได้แก่ ชนิดตัวแปร การรับและแสดงผล การตัดสินใจ การทำซ้ำ แถวลำดับ ฟังก์ชันต่าง ๆ และการสร้างฟังก์ชันใหม่ โดยผลลัพธ์ของโปรแกรมตัวอย่างจะแสดงผลที่คอนโซล หรือหน้าต่างของ REPL ทั้งนี้เพื่อให้ผู้อ่านได้เรียนรู้ ทดลอง และดูผลลัพธ์จากการสั่งงานโดยไม่กังวลเรื่องการต่อวงจรกับอุปกรณ์ภายนอก

เมื่อจบบทนี้ผู้อ่านจะเข้าใจถึงพื้นฐานของภาษาไพธอน รวมทั้งสามารถประยุกต์ใช้คำสั่งภาษาไพธอนเพื่อเขียนโปรแกรมและควบคุมสั่งงานอุปกรณ์ฝังตัวได้

ภาษาไพธอน

ภาษาไพธอนเป็นภาษาสคริปต์ที่ออกแบบมาเพื่อใช้งานทั่วไป ทำให้ง่ายต่อการศึกษาเรียนรู้ แต่สิ่งที่ต้องทำความเข้าใจก่อนฝึกเขียนโปรแกรมภาษานี้คือ การจัดตำแหน่งของชุดคำสั่ง ทั้งนี้เนื่องจากภาษาไพธอนไม่มีเครื่องหมายใช้แทนการเริ่มบล็อกและปิดบล็อกคำสั่งเหมือนภาษาทั่วไป ตัวอย่างเช่น การเขียนโปรแกรมภาษา C เพื่อตรวจสอบตัวแปร x ถ้าตัวแปร x มีค่า 0-59 ให้แสดงค่า F ถ้ามีค่าอยู่ในช่วง 60-79 ให้แสดง P และถ้าอยู่ในช่วง 80-100 ให้แสดงค่า G แต่ถ้าเป็นกรณีอื่น ๆ ให้แสดงข้อความ Error สามารถเขียนได้ดังนี้

```
if ((x>=0) && (x<=59)) {
      print(“Your score = %d\n”, x);
      print(“Your grade = F\n”);
} else
if ((x>=60) && (x<=79)) { print(“Your score = %d\n”, x);
      print(“Your grade = P\n”) } else
if ((x>=80) && (x<=100)) {
      print(“Your score = %d\n”, x); print(“Your grade = G\n”);
} else {print(“Your score = %d\n”, x);print(“Your grade = Error\n”);}
```

จากโค้ดด้านบนจะพบว่าเครื่องหมาย { และ } เป็นตัวบอกบล็อกของการทำงานว่าคำสั่งต่าง ๆ ภายใต้เครื่องหมายนี้เป็นของเงื่อนไขใด และเครื่องหมาย ; ใช้สำหรับ

แทนการบอกการสิ้นสุดคำสั่งแต่ละคำสั่ง นอกจากนี้ผู้เขียนโปรแกรมสามารถเขียนให้แต่ละเงื่อนไขจัดเรียงพิมพ์แตกต่างกัน เช่น สามารถเขียนคำสั่งหลายคำสั่งเรียงในบรรทัดเดียวกัน สามารถแยก else กับ if ไปอยู่ต่างบรรทัดกัน เป็นต้น เหตุการณ์เหล่านี้จะไม่สามารถกระทำได้ในภาษาไพธอน เนื่องจาก รูปแบบการเขียนคำสั่งของไพธอนใช้หลักของการเขียนบรรทัดละ 1 คำสั่ง และคำสั่งที่อยู่บล็อกทำงานเดียวกันต้องอยู่ในระดับกั้นหน้าที่เท่ากัน ด้วยเหตุนี้จึงนิยมใช้ Tab แทนการย่อหน้าในแต่ละระดับ ดังนั้นโค้ดภาษา C ด้านบนเมื่อเขียนเป็นภาษาไพธอนจึงมีลักษณะดังต่อไปนี้

```
if (x>=0) and (x<=59):
     print("Your score = ",x)
     print("Your grade = F")
elif (x>=60) and (x<=79):
     print("Your score = ",x)
     print("Your grade = P")
elif (x>=80) and (x<=100):
     print("Your score = ",x)
     print("Your grade = G")
else:
     print("Your score =",x)
     print("Your grade = Error")
```

ตัวแปร

ตัวแปรคือพื้นที่ที่ถูกจองในหน่วยวามจำ เพื่อให้ค่าต่าง ๆ สามารถนำกลับมาใช้ได้อีก เมื่อมีการสร้างตัวแปรพื้นที่ในหน่วยความจำจะถูกจอง โดยขนาดที่ถูกจองนั้นขึ้นอยู่กับชนิดของตัวแปร ในตัวแปรสามารถใส่ค่าได้ทั้งข้อความและตัวเลขหรืออื่น ๆ

ตัวแปร (Variable) เป็นวิธีการอ้างอิงชื่อเรียกแทนค่าตำแหน่งจริงของข้อมูลที่ถูกจัดเก็บในหน่วยความจำประเภทแรมของไมโครคอนโทรลเลอร์ ขั้นตอนของการสร้างตัวแปร คือ ร้องขอพื้นที่หน่วยความจำที่ว่างจากการใช้งานในปริมาณที่ระบุ (ขึ้นอยู่กับประเภทของข้อมูล) ถ้ามีพื้นที่หน่วยความจำเพียงพอจึงได้รับค่าตำแหน่งเริ่มต้นเก็บข้อมูลที่ร้องขอไปและดำเนินการโอนข้อมูลที่กำหนดไว้ไปเก็บหน่วยความจำนั้น แต่ถ้าหน่วยความจำมีพื้นที่ไม่มากพอจะไม่สามารถสร้างตัวแปรนั้นได้ ซึ่งค่าตำแหน่งของหน่วยความจำนั้นมีค่าที่ไม่แน่นอนขึ้นอยู่กับตำแหน่งว่าง ดังนั้น จึงใช้หลักการเรียกชื่อแทนค่าตำแหน่งของหน่วยความจำเพื่อให้ง่ายสำหรับผู้เขียนโปรแกรม และผู้ที่ทำหน้าที่แปลงชื่อและค่าตำแหน่งไปมาจึงตกเป็นของระบบปฏิบัติการ หรือตัวรันไทม์ไลบรารี (Run-time library)

ภาษาไพธอนมีชนิดตัวแปรที่แตกต่างจากภาษา C/C++ หรือ Java และในภาษาไพธอนไม่จำเป็นต้องประกาศชนิดตัวแปรเพื่อระบุประเภทของตัวแปรก่อนใช้ ดังนั้นในการสร้างตัวแปรผู้เขียนจึงต้องระมัดระวังในเรื่องการแปลงค่าระหว่างตัวแปร และ

การตั้งชื่อของตัวแปรเนื่องจากภาษาไพธอนเป็นภาษาแบบเคสเซ็นซิทีฟ (case sensitive) คือ ตัวอักษรพิมพ์เล็กถูกมองเป็นคนละตัวกับตัวอักษรพิมพ์ใหญ่ ดังนั้น เมื่อเขียนโค้ดให้กำหนด a มีค่าเป็น 10 และสั่งแสดงค่า A จะให้ผลดังนี้ ทั้งนี้เนื่องจากการสั่งแสดงผลค่าที่เก็บในตัว A ไม่สามารถดำเนินการได้ เพราะชื่อ 'A' ไม่ได้ถูกกำหนดมาก่อนที่เรียกบรรทัดนี้

```
>>> a=10
>>> print(A)
  Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
  NameError: name 'A' isn't defined
```

หลักการตั้งชื่อตัวแปร

ชื่อของตัวแปรต้องถูกใช้ในการอ้างอิงร่วมกับคำสั่งทั้งหมดในโปรแกรม ดังนั้น การตั้งชื่อให้กับตัวแปรจึงต้องมีหลักการที่รัดกุม เพื่อให้ภาษาไพธอนสามารถประมวลผลได้โดยราบลื่น และเพื่อให้โค้ดสามารถอ่านได้ง่าย

1. กฎในการตั้งชื่อตัวแปร

การตั้งชื่อตัวแปรจะต้องดำเนินการตามภายใต้กฎดังต่อไปนี้

1) ขึ้นต้นด้วยภาษาอังกฤษหรือเครื่องหมายอันเดอร์สกอร์ ตามด้วยอักษรพิมพ์ใหญ่ พิมพ์เล็ก ตัวเลข หรือ เครื่องหมายอันเดอร์สกอร์เท่านั้น
2) ห้ามเว้นวรรคและห้ามอักขระพิเศษต่าง ๆ
3) อักษรพิมพ์ใหญ่และอักษรพิมพ์เล็กต่างกัน
4) ตั้งชื่อให้มีความหมายสอดคล้องกับข้อมูล
5) ห้ามซ้ำกับคำสงวน

2. คำสงวน

คำสงวนคือคำที่ภาษาโปรแกรมสงวนไว้ใช้ในคำสั่งพื้นฐานต่าง ๆ ซึ่งเป็นคำสั่งที่สามารถใช้งานได้ทันที เนื่องจากชื่อของตัวแปรถูกกำหนดได้โดยตรง จึงทำให้มีโอกาสซ้ำกับคำสงวนได้ หากชื่อตัวแปรซ้ำกับคำสงวนจะทำให้ตัวแปรภาษาสับสนได้

```
and       assert   break    class
continue  def      del      elif
else      except   exec     finally
for       from     global   if
import    in       is       lamda
not       or       pass     print
raise     return   try      while    yield
```

การกำหนดค่าให้กับตัวแปร

ในภาษาไพธอนสามารถกำหนดค่าให้กับตัวแปรด้วยเครื่องหมายเท่ากับ โดยแบ่งเป็นสองด้านคือ ด้านซ้ายของเครื่องหมายเท่ากับคือตัวแปรที่ต้องการกำหนดค่า และด้านขวาของเครื่องหมายเท่ากับคือค่าที่ต้องการใส่ลงไปในตัวแปร ซึ่งสามารถเป็นได้ทั้งตัวเลข ข้อความ ข้อมูลชนิดต่าง ๆ หรือข้อมูลในตัวแปรอื่น ๆ

วิธีการกำหนดค่าให้กับตัวแปรมีรูปแบบ ดังนี้

L = R

โดย L คือ ชื่อของตัวแปร

R คือ ค่าของตัวแปร หรือนิพจน์

ตัวอย่างการกำหนดสตริง "micro:bit" ในตัวแปร BoardName เขียนเป็นชุดคำสั่งได้ดังนี้

```
BoardName = "micro:bit"
```

หรือ

```
BoardName = 'micro:bit'
```

ตัวอย่างการนำค่า 3.1415926 เก็บในตัวแปรชื่อ MyPi เขียนเป็นชุดคำสั่งได้ดังนี้

```
MyPi = 3.1415926
```

สำหรับการสร้างตัวแปรและไม่กำหนดให้ตัวแปรเก็บค่าประเภทใด แต่ต้องการประกาศให้ตัวแปลภาษารู้จักชื่อตัวแปรนั้นก่อนสร้างขึ้นจริง สามารถกระทำได้โดยกำหนดค่าให้ตัวแปรมีค่าเป็น None ดังนี้

ชื่อตัวแปร = None

ตัวอย่างการประกาศตัวแปรและกำหนดค่าให้กับตัวแปรแบบอื่น ๆ เป็นดังนี้

```
myNum = 12345
myStr = "ETT"
myList = ["Hello", "World"]
myTuple = ("mp”, "thon", 2020)
myDict = {"title" : "mpthon"}
myNum2 = myNum
```

การตรวจสอบประเภทของตัวแปร

บทที่

29

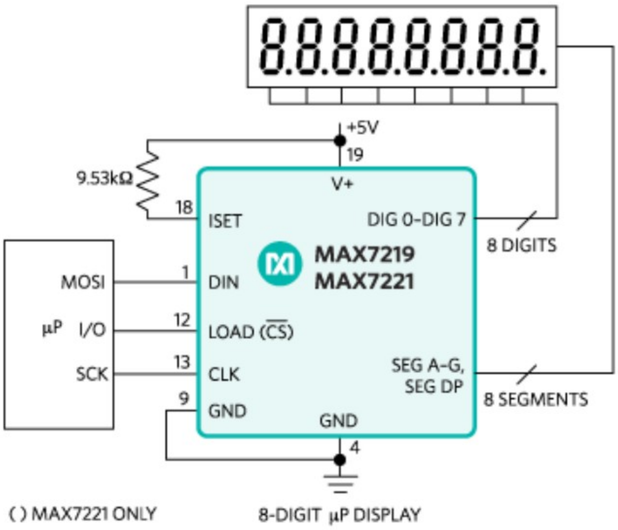
การใช้งาน MAX7219 Dot Matrix 8x32 Display

ในบทนี้กล่าวถึงการเขียนโปรแกรมด้วยภาษาไพธอนเพื่อเชื่อมประสานกับบอร์ด MAX7219 Dot Matrix 8x32 Display สำหรับควบคุมการแสดงผลหลอดแอลอีดีในแบบเมตริกซ์หรือหลายหลอดเรียงกันเป็นดังภาพที่ 29-1 โดยใช้การสื่อสารผ่านบัส SPI จาก micro:bit ผ่านการขยายพอร์ตด้วย ET-BBC GPIO V1 เพื่อสั่งงานการแสดงผลแอลอีดีทั้ง 4 หลัก หรือ 8x32 หลอด เมื่อจบบทนี้ ผู้อ่านจะเข้าใจถึงขั้นตอนการเขียนโปรแกรม และวิธีการเขียนโปรแกรมรับข้อมูลนำเข้าข้อมูลในการสื่อสารกับไอซี MAX7219 เพื่อควบคุมการแสดงผลของแผงแอลอีดี โดยตัวอย่างกล่าวถึงการแสดงอักษร การสุ่มการแสดงการติดของหลอดแอลอีดี และการสั่งงานปรับความสว่างของหลอด อันเป็นพื้นฐานสำหรับการประยุกต์ต่อไป

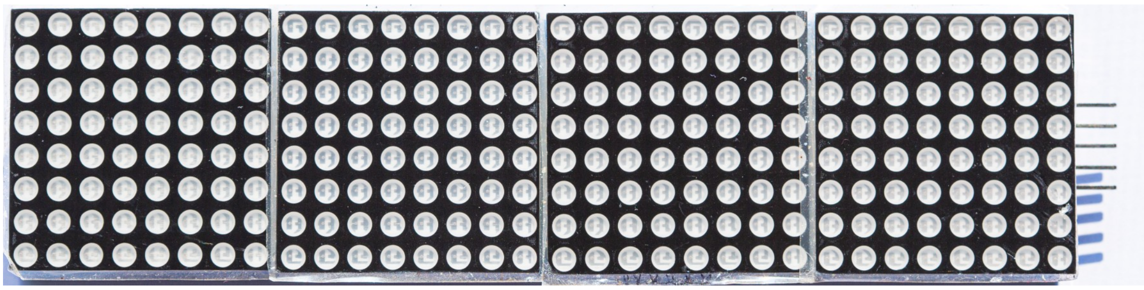
แนะนำบอร์ด MAX7219 Dot Matrix 8x32 Display

ไอซี MAX7219 เป็นไอซีที่ออกแบบโดยบริษัท maxim integrated เพื่อใช้สำหรับขับวงจรของ 7-Segment ได้สูงสุด 8 หลัก ดังภาพที่ 29-1 แต่ในโมดูล MAX7219 Dot Matrix 8x32 Display เลือกใช้บอร์ดแสดงผลแอลอีดีที่จัดเรียงขาใหม่ เพื่อให้ได้ผลลัพธ์เป็นแอลอีดีขนาด 8x8 หลอดต่อ 1 หน่วย และเมื่อนำมาต่อกัน 4 ตัวจึงสามารถแสดงผลเมตริกซ์แอลอีดีขนาด 8x32 หลอด โดยใช้ไอซีเพียงตัวเดียว

บอร์ด MAX7219 Dot Matrix 8x32 Display (ภาพที่ 29-2) เป็นโมดูลที่ติดตั้งไอซี MAX7219 และแผงหลอดแอลอีดีแบบเมตริกซ์ขนาด 8x8 หลอดจำนวน 4 หลัก ทำให้สามารถแสดงผลข้อมูลได้ 8x32 หลอด โดยใช้บัสสื่อสารแบบ SPI ที่ความเร็วในการทำงานสูงสุด 10MHz ใช้กระแสไฟ 150uA ทำงานกับแรงดันไฟฟ้ากระแสตรง 4V ถึง 5.5V รองรับการขับหลอดแอลอีดีแบบแคโธดร่วม สามารถประยุกต์ใช้ควบคุมการแสดงผลแบบกราฟแท่ง ด็อตเมตริกซ์แอลอีดี หรือมิเตอร์ในแผงควบคุม เป็นต้น



ภาพที่ 29-1 ผังการต่อ MAX7219 กับ 7-Segment 8 หลัก
ที่มา https://www.maximintegrated.com/en/products/power/display-power-control/MAX7219.html

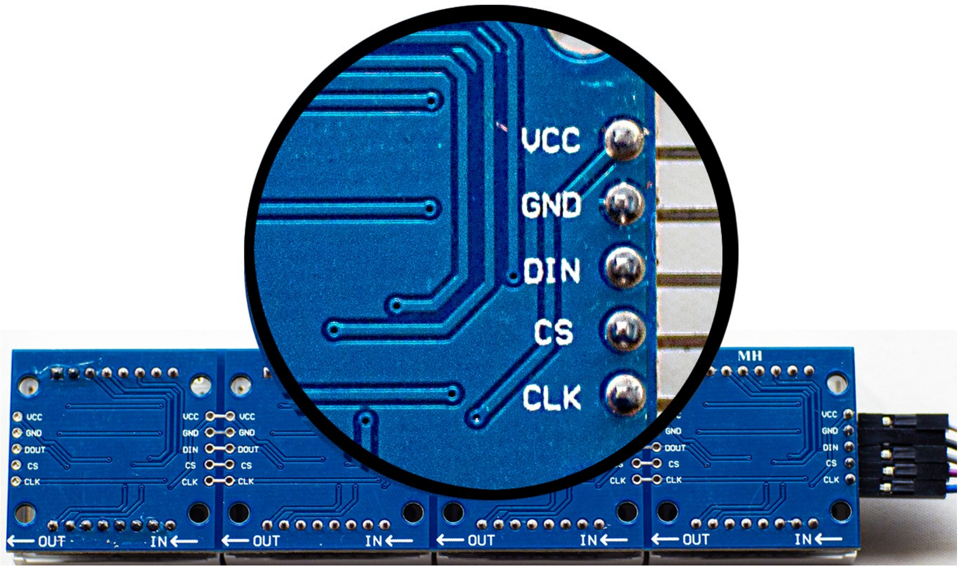


ภาพที่ 29-2 บอร์ด MAX7219 Dot Matrix 8x32 Display

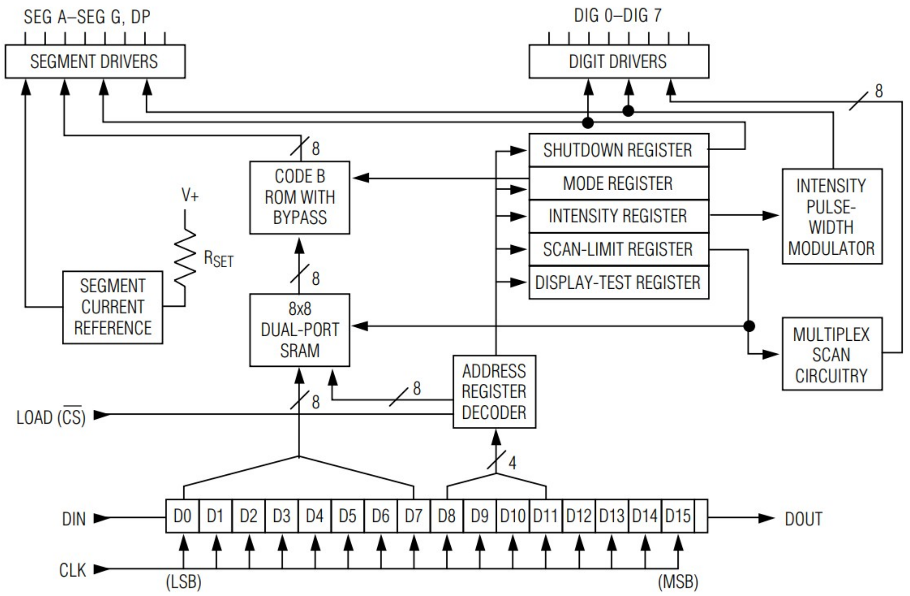
การเชื่อมต่อและหลักการทำงาน

หลักการทำงานของ MAX7219 Dot Matrix 8x32 Display คือ ผู้เขียนโปรแกรมทำการส่งข้อมูลสำหรับเปิด/ปิดการทำงานของหลอดแอลอีดีขนาด 8x8 หลอด โดยมีทั้งสิ้น 4 หลัก ดังนั้น จึงต้องส่งข้อมูล 4 ครั้งเพื่อให้ข้อมูลถูกผลักส่งไปยังหลักถัด ๆ ไป การเชื่อมต่อกับ MAX7219 ต้องอาศัยบัส SPI ที่มีการเรียงขาไว้ดังภาพที่ 29-3

ผังการทำงานของ MAX7219 เป็นดังภาพที่ 29-4 จะพบว่าการส่งข้อมูลจะถูกส่งไปครั้งละ 16 บิต โดยบิตที่ 0 ถึง 7 เป็นข้อมูล และบิตที่ 8 ถึง 11 เป็นค่าตำแหน่งของเรจิสเตอร์ที่ต้องการส่งข้อมูลให้ โดยบิตที่ 12 ถึง 15 เป็นบิตที่ไม่ได้ถูกใช้งาน โดยค่าตำแหน่งของเรจิสเตอร์เป็นดังตารางที่ 29-1



ภาพที่ 29-3 การเรียงขาของ MAX7219 Dot Matrix 8x32 Display

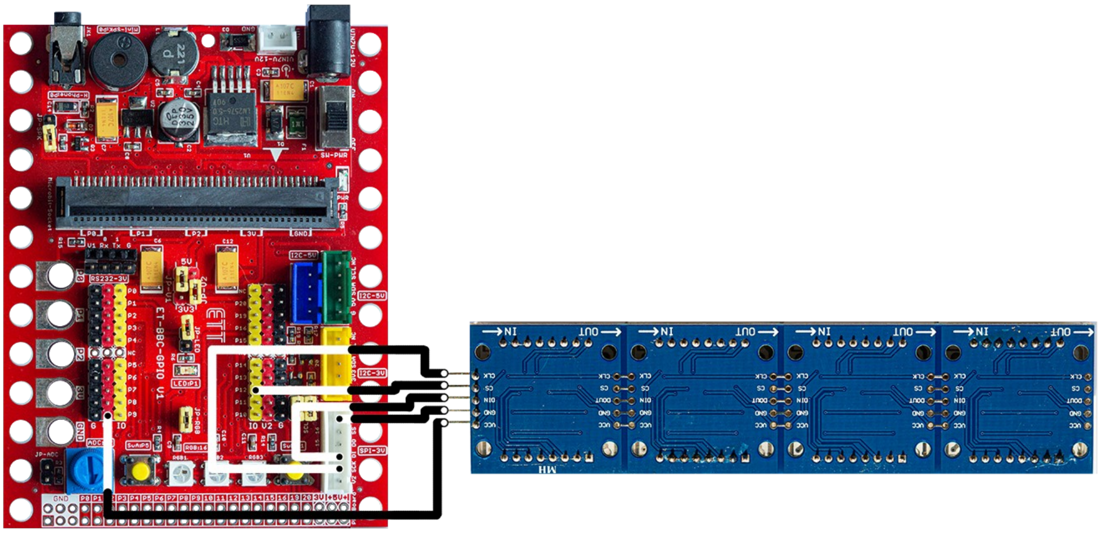


ภาพที่ 29-4 ผังการการทำงานของ MAX7219

การต่อเพื่อสั่งงาน MAX7219 Dot Matrix 8x32 Display ผ่านทาง ET-BBC GPIO V1 คือ ขา VCC ของบอร์ดต่อเข้ากับแหล่งจ่ายไฟจากบอร์ด ET-BBC GPIO V1 ขา GND ต่อกับกับกราวด์ ขา DIN ต่อกับเข้า DI บนขั้วต่อ SPI หรือ Pin14 ขา CS ต่อเข้ากับ Pin12 ขา CLK ต่อกับขา SCK บนขั้วต่อ SPI หรือ Pin13 ดังภาพที่ 29-5

ตารางที่ 29-1 ค่าตำแหน่งเรจิสเตอร์ของ MAX7219

บิต								รหัสฐาน 16	ความหมาย
D15	D14	D13	D12	D11	D10	D9	D8		
X	X	X	X	0	0	0	0	0xX0	No-op
X	X	X	X	0	0	0	1	0xX1	Digit 0
X	X	X	X	0	0	1	0	0xX2	Digit 1
X	X	X	X	0	0	1	1	0xX3	Digit 2
X	X	X	X	0	1	0	0	0xX4	Digit 3
X	X	X	X	0	1	0	1	0xX5	Digit 4
X	X	X	X	0	1	1	0	0xX6	Digit 5
X	X	X	X	0	1	1	1	0xX7	Digit 6
X	X	X	X	1	0	0	0	0xX8	Digit 7
X	X	X	X	1	0	0	1	0xX9	Decode Mode
X	X	X	X	1	0	1	0	0xXA	Intensity
X	X	X	X	1	0	1	1	0xXB	Scan Limit
X	X	X	X	1	1	0	0	0xXC	Shutdown
X	X	X	X	1	1	1	1	0xXF	Display Test



ภาพที่ 29-5 การเชื่อมต่อ MAX7219 Dot Matrix 8x32 Display กับ ET-BBC GPIO V1

ตัวอย่างโปรแกรมแสดงอักขระ

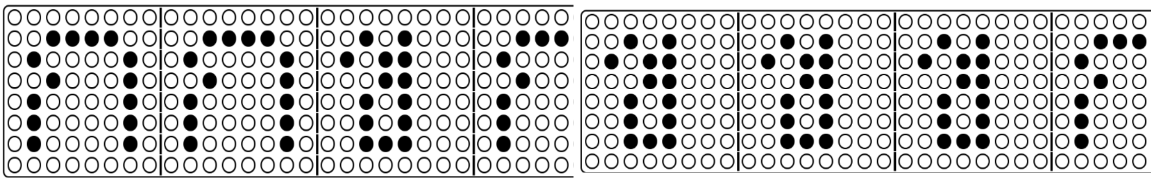
ตัวอย่างโปรแกรมการแสดงอักขนะ "ก" และ "ข" แบบสุ่มในแต่ละหลักเมื่อผู้ใช้กด button_a ของ micro:bit และสิ้นสุดการทำงานเมื่อกด button_b เป็นดังตัวอย่างโปรแกรม max7912-01.py และตัวอย่างผลลัพธ์การทำงานเป็นดังภาพที่ 29-6

ตัวอย่างโปรแกรม max7912-01.py

```
import microbit as mb
import random
khorkhai =
bytearray([0b00000000,0b00101000,0b01011000,0b00011000,0b00101000,
0b00101000,0b00111000,0b00000000])
korkai =
bytearray([0b00000000,0b00111100,0b01000010,0b00100010,0b01000010,
0b01000010, 0b01000010,0b00000000])
CMD_NOOP = 0
CMD_DIGIT0 = 1
CMD_DECODE_MODE  = 9
CMD_INTENSITY = 10
CMD_SCAN_LIMIT = 11
CMD_SHUTDOWN = 12
CMD_DISPLAY_TEST = 15
cs_pin = mb.pin12
max7219_buffer = [0 for i in range(32)]
chars = [korkai, khorkhai]
def max7219_write( cmd, data ):
    cs_pin.write_digital(0)
    mb.spi.write( bytearray([cmd, data]) ) # 3
    mb.spi.write( bytearray([cmd, data]) ) # 2
    mb.spi.write( bytearray([cmd, data]) ) # 1
    mb.spi.write( bytearray([cmd, data]) ) # 0
    cs_pin.write_digital(1)
def max7219_init():
    global max7219_buffer
    mb.spi.init()
    for cmd, data in ((CMD_SHUTDOWN,0),(CMD_DISPLAY_TEST,0),
(CMD_INTENSITY,12),(CMD_SCAN_LIMIT,7),(CMD_DECODE_MODE,0),
(CMD_SHUTDOWN,1)):
```

```
        max7219_write(cmd,data)
    max7219_clear()
def max7219_shutdown():
    max7219_write(CMD_SHUTDOWN, 0)
def max7219_clear():
    global max7219_buffer # ส่ง 8 Line, Line ละ 4 คอลัมน์
    for line in range(8):
        line_no = CMD_DIGIT0+line
        for col in range(4):
            idx = col*8+line
            max7219_buffer[idx] = 0
def max7219_putc(idx,ch): #นำข้อมูลไปใส่ใน max7219_buffer[idx*8]
    global max7219_buffer
    addr = idx*8 # idx = 0,1,2,3
    for i in range(8):
        max7219_buffer[addr+i]=ch[i]
def max7219_show():
    global max7219_buffer # ส่ง 8 Line, Line ละ 4 คอลัมน์
    for line in range(8):
        line_no = CMD_DIGIT0+line
        cs_pin.write_digital(0)
        for col in range(4):
            idx = col*8+line
            data = max7219_buffer[idx]
            mb.spi.write(bytearray([line_no, data]))
        cs_pin.write_digital(1)
def demo1():
    for i in range(4):
        max7219_putc(i, random.choice(chars))
if __name__=="__main__":
    max7219_init()
    demo1()
    while True:
        if mb.button_a.is_pressed():
            demo1()
        if mb.button_b.is_pressed():
```

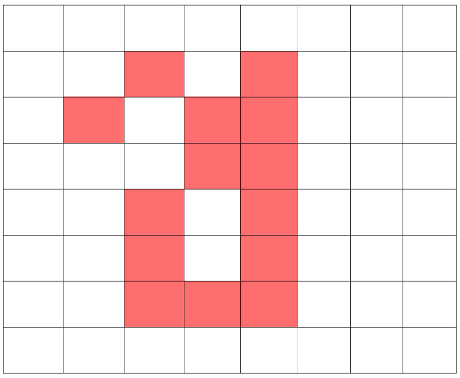
No.	Code
62	`            break`
63	`        max7219_show()`
64	`    max7219_shutdown()`



ภาพที่ 29-6 ผลของการทำงานตามตัวอย่าง max7912-01.py

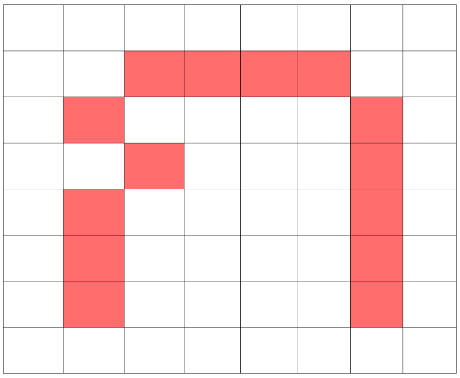
จากตัวอย่างโปรแกรม max7912-01.py จะพบว่าในบรรทัดที่ 3 ได้สร้างตัวแปร khorkhai ให้เป็น bytearray เพื่อเก็บค่าบิตของตัวอักษร 'ข' ดังนี้

00000000_2
00101000_2
01011000_2
00011000_2
00101000_2
00101000_2
00111000_2
00000000_2



ในบรรทัดที่ 4 สร้างตัวแปร korkai สำหรับเก็บข้อมูลบิตของตัวอักษร 'ก' ดังนี้

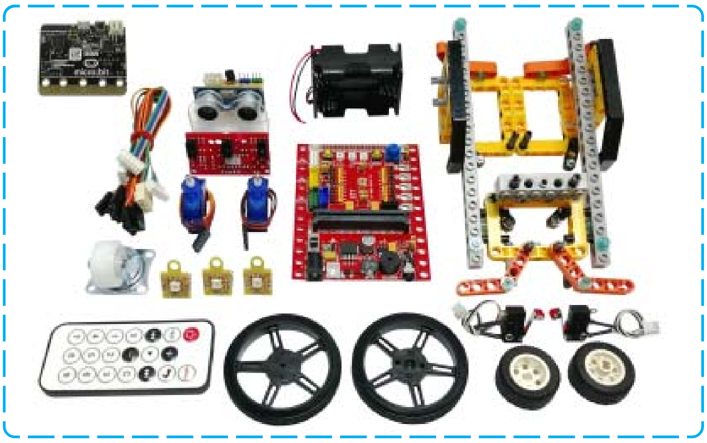
00000000_2
00111100_2
01000010_2
00100010_2
01000010_2
01000010_2
01000010_2
00000000_2



ET-BBC ROBOT KIT + MICRO BIT (P-ET-A-00640)

ET-BBC ROBOT KIT ... เป็น ROBOT ประเภทสามล้อ ประกอบสร้างด้วยชิ้นส่วน TECHNIC BRICKS ทำให้สามารถดัดแปลงประกอบเป็น ROBOT หรือ PROJECT อื่นๆ ได้โดยง่าย ใช้ตัวควบคุม BBC MICRO:BIT ในการควบคุม เขียนโปรแกรมในรูปแบบ BLOCK ง่ายต่อการใช้งาน เรียนรู้

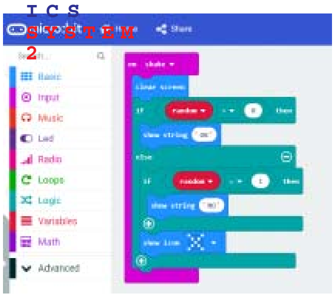
ใช้บอร์ด ET-BBC GPIO เป็นส่วนวงจรขยายการทำงาน ของบอร์ด BBC MICRO:BIT ในการควบคุมวงจรต่างๆ เช่น SERVO MOTOR แบบหมุน 360 องศา ตัว SENSOR TRACKER



เข้าดูรายละเอียดเพิ่มเติมได้ที่

ET-ROBOT-ICS SYSTEM

micro:bit

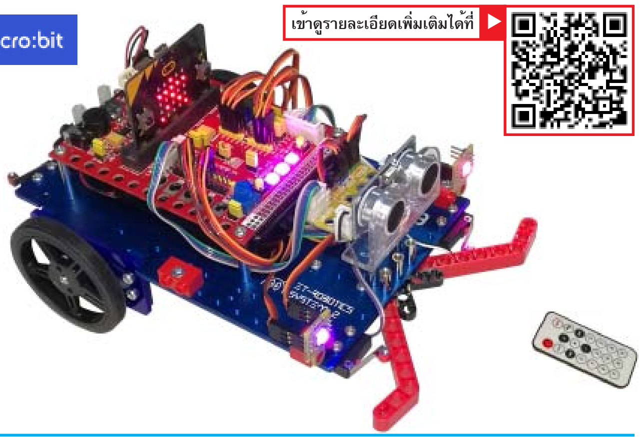


ET-BBC ROBOT + MICRO BIT (P-ET-A-00638)

ET-BBC ROBOT ... จะเป็น ROBOT ที่มีส่วนประกอบต่างๆ เช่นเดียวกับ ET-BBC ROBOR KIT โดยจะแตกต่างในส่วนโครงสร้างของตัว ROBOT จากชิ้นส่วน TECHNIC BRICKS เปลี่ยนเป็นใช้ ALUMINUN ชุบ ANODIAED สีน้ำเงินแทน

micro:bit

เข้าดูรายละเอียดเพิ่มเติมได้ที่



BBC MICRO:BIT SET A (C-YA-A-00291)

BBC MICRO:BIT SET A เป็นบอร์ดไมโครคอนโทรลเลอร์ขนาดเล็กที่จัดเป็นชุดพร้อมใช้งาน ประกอบด้วย **1.** บอร์ด BBC MICRO:BIT **2.** กล่องพลาสติกใส่บอร์ด **3.** CABLE USB MICRO **4.** ลังถ่าน AAA X 2 **5.** ถ่าน AAA จำนวน 2 ก้อน **6.** เอกสารใช้งาน

บอร์ด BBC MICRO:BIT เป็นบอร์ดไมโครคอนโทรลเลอร์ขนาดเล็ก ถูกออกแบบสร้างโดย บริษัทแพร่ภาพกระจายเสียงอังกฤษ หรือ BBC เพื่อสนับสนุนการเรียนการสอนใช้งานคอมพิวเตอร์ขนาดเล็ก ให้กับเด็กๆ มีภาษาสำหรับเด็กในการเขียนโปรแกรม เช่น ภาษาในรูปแบบการต่อแบบ BLOCK หรือภาษาที่สูงขึ้น เช่น JAVA SCRPT, ภาษาไพธอน ฯลฯ

เข้าดูรายละเอียดเพิ่มเติมได้ที่

เรียนรู้เข้าใจใช้งาน BBC MICRO:BIT
ด้วยภาษา MICRO PYTHON
P/N A-BK-P-00153

A-BK-P-00153

เรียนรู้เข้าใจใช้งาน BBC MICRO:BIT
ด้วยภาษา MICRO PYTHON
ISBN 978-616-92864-1-7

9 786169 286417

บริษัท อีทีที จำกัด
1112/96-98 ถนนสุขุมวิท แขวงพระโขนง เขตคลองเตย กรุงเทพฯ 10110
1112/96-98 Sukhumvit Rd., Phrakanong Klongtoey Bangkok 10110
Tel : 02-7121120 Fax : 02-3917216

http://www.etteam.com
LINE ID : etteam1
: facebook.com/ett.co.th
e-mail : sale@etteam.com