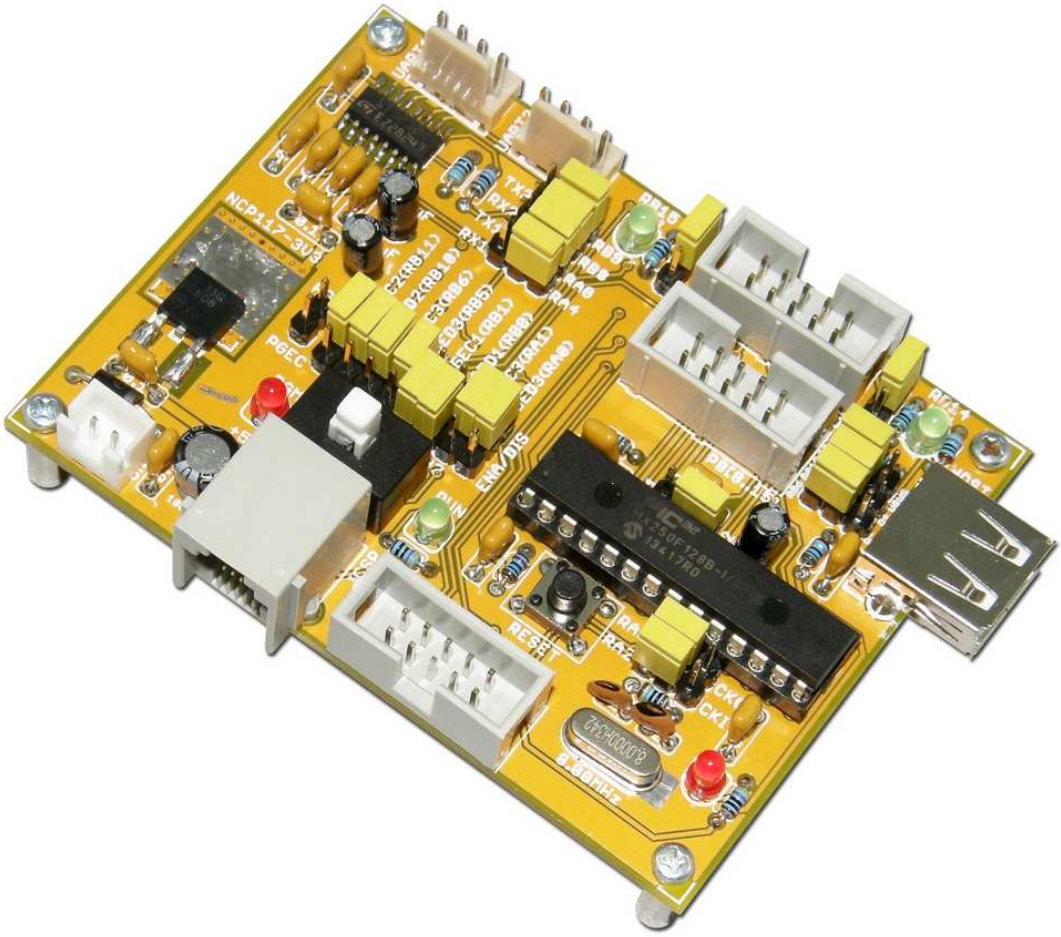
ET-BASE PIC32MX250F128B



ET-BASE PIC32MX250F128B เป็นบอร์ดไมโครคอนโทรลเลอร์ขนาด 32บิต ตระกูล PIC32 Core MIPS32 MK4 ของ Microchips โดยเลือกใช้ชิพ MCU เบอร์ PIC32MX250F128B เป็น MCU ประจำบอร์ด โดย MCU เบอร์นี้เป็น MCU ขนาด 32บิต ที่บรรจุอยู่ในตัวถังแบบ SPDIP ขนาด 28PIN ซึ่งถึงแม้ว่าจะเป็น MCU ที่มีตัวถังขนาดเล็กแต่ภายในได้บรรจุศักยภาพด้านต่างๆ รวมไว้มากมาย สามารถนำไปประยุกต์ใช้งานในรูปแบบต่างๆได้โดยง่าย

จากการสนับสนุนของทาง Microchip ในด้านของเครื่องมือที่ใช้ในการพัฒนาโปรแกรม ทั้ง Text Editor และ Compiler รวมทั้ง Library และตัวอย่างต่างๆ ซึ่งมีออกมาสนับสนุนมากมาย ทำให้ผู้พัฒนาสามารถค้นหาข้อมูลและตัวอย่างด้านต่างๆ สำหรับนำมาเป็นต้นแบบแนวทางหรือข้อมูลสำหรับใช้พัฒนาโปรแกรมและแก้ไขดัดแปลงต่อยอดไปสู่ชิ้นงานจริงได้โดยง่าย ซึ่งจะช่วยให้แนวทางในการพัฒนาโปรแกรมมีความสะดวกรวดเร็วและประหยัดเวลาและงบประมาณที่ต้องใช้ในการพัฒนาโปรแกรมได้มากขึ้นด้วย

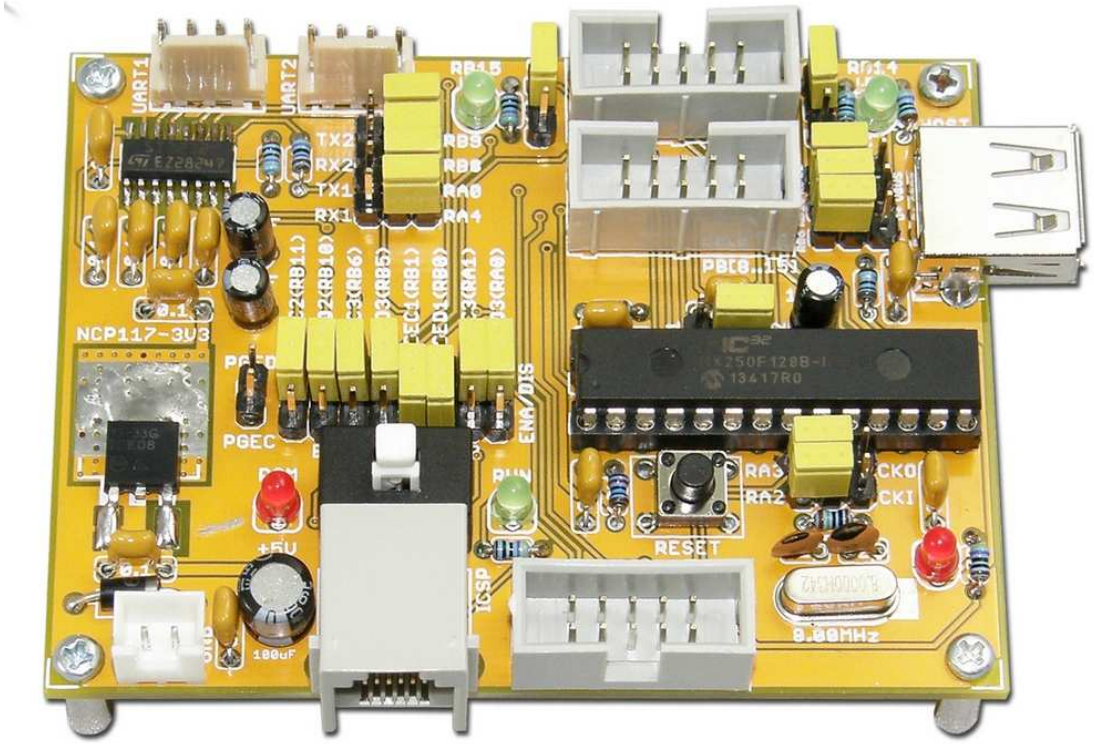
คุณสมบัติของ MCU เบอร์ PIC32MX250F128B

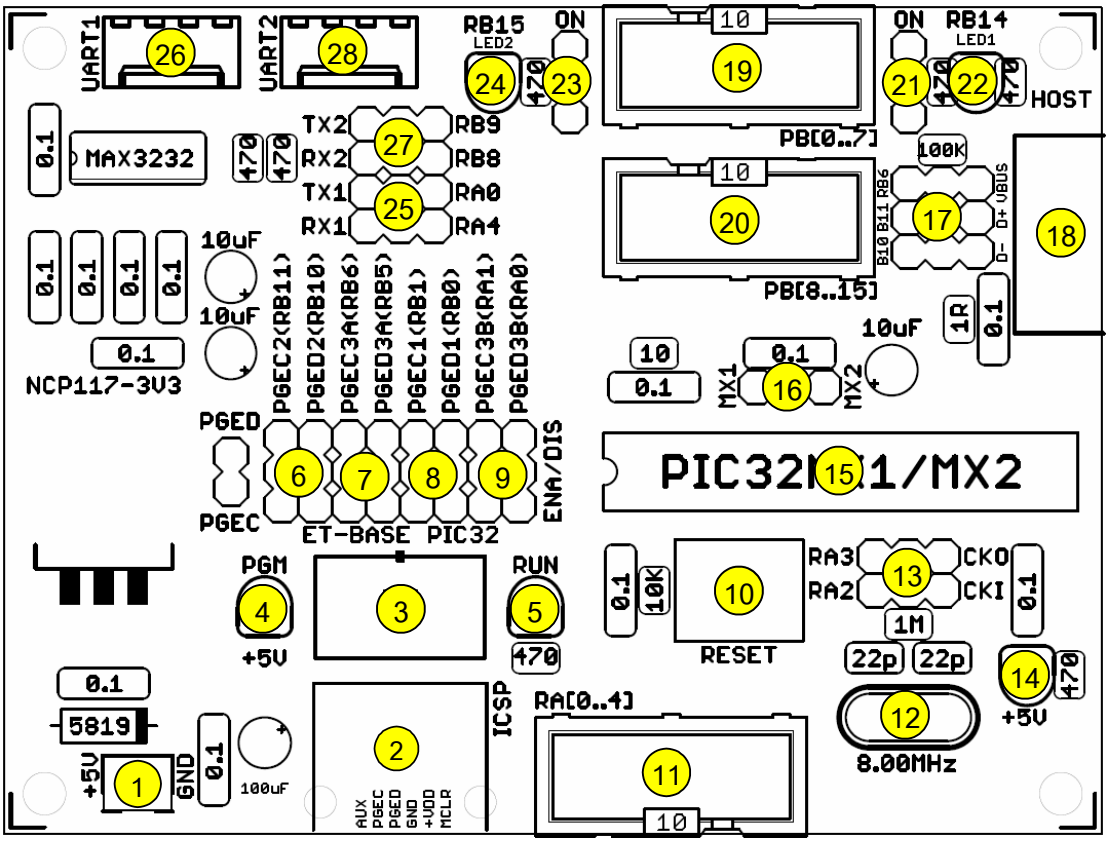
- 32Bit Core 50MHz/83DMIPS MIPS32 M4K
  - Internal Oscillator FRC 8MHz (0.9% Error)
  - Programmable Phase Lock Loop
  - Watchdog Timer
- 128KByte Flash(Program Memory) และ 3KByte Flash(Boot Flash Memory)
- 32KByte SRAM(Data Memory)
- 21 I/O Pin
  - 19 Pin Remappable I/O
  - 5 Timer/ 5Capture/ 5Compare
  - 2 Channel UART
  - 2 Channel SPI
  - 5 Channel External Interrupt
  - 3 Channel Analog Comparator
  - 2 Channel I2C
  - 4 Channel DMA
  - 9 Channel 10Bit(1Msps) ADC
  - 1 Channel USB(Host/Device/OTG)
- SPDIP28 Package
- 2.3V to 3.6V, -40ºC to +85ºC, DC to 50 MHz
- 2.3V to 3.6V, -40ºC to +105ºC, DC to 40 MHz

คุณสมบัติของ บอร์ด ET-BASE PIC32MX250F128B

- รองรับ MCU PIC32MX250F128B แบบ 28Pin SPDIP พร้อม Socket แบบขากลม สามารถถอดเปลี่ยน MCU ใหม่ได้โดยง่าย
- 2 ช่อง UART RS232 พร้อม Jumper ตัดต่อสัญญาณใช้งานหรือไม่ใช้งานได้ตามต้องการ
  - UART1 ใช้ RA4 เป็น RX1 และ RA0 เป็น TX1
  - UART2 ใช้ RB8 เป็น RX2 และ RB9 เป็น TX2
- 2 LED Output พร้อม Jumper ตัดต่อสัญญาณใช้หรือไม่ใช้ได้ตามต้องการ
  - LED1 ใช้ RB14 ควบคุมการติดดับ(Active High)
  - LED2 ใช้ RB15 ควบคุมการติดดับ(Active High)
- 8MHz Crystal Oscillator พร้อม Jumper เพื่อเลือกใช้ Crystal หรือ Internal RC ภายใน
- 1 Switch Reset แบบ Push Button
- 1 Switch และ Jumper สำหรับเลือกสลับสัญญาณเพื่อใช้เป็น ICSP และ GPIO
  - Jumper สำหรับเลือกชุดสัญญาณ ICSP ได้อิสระทั้ง 3ชุด เพื่อให้เปลี่ยนได้ตามต้องการในกรณีสัญญาณ ICSP ทับซ้อนกับฟังก์ชั่นที่ต้องใช้งาน
  - มี LED แสดงสถานะ Program(PGM) และ Run(RUN) แสดงให้ทราบตำแหน่งของ Switch ที่เลือก
- 3 ชุด 10Pin IDE แบบ Header Block สำหรับใช้เป็นจุดเชื่อมต่อสัญญาณ GPIO ใช้งาน RA[0..4], RB[0..7] และ RB[8..15]
- 1 Port USB Host พร้อม Jumper ตัดต่อสัญญาณเพื่อเลือกใช้งานหรือไม่ใช้งานได้
- 5VDC Power Input พร้อม Regulate 3.3V/1A พร้อม LED แสดงสถานะ
- ขนาด PCB 8.2 x 6.2 ซม

โครงสร้างของบอร์ด





รูปแสดง โครงสร้างของบอร์ด ET-BASE PIC32MX

- **หมายเลข 1** คือ ขั้วต่อแหล่งจ่ายไฟเลี้ยงวงจรของบอร์ด ใช้กับแหล่งจ่ายไฟ +5VDC
- **หมายเลข 2** คือ ขั้วต่อ ICSP สำหรับใช้เชื่อมต่อกับเครื่องโปรแกรมและดีบักตามมาตรฐาน ICSP ของ Microchips
- **หมายเลข 3** คือ สวิตช์ สำหรับเลือกโหมดการทำงานระหว่าง Run(RUN) และ Program(PGM)
- **หมายเลข 4** คือ LED สีแดง แสดงสถานะ PGM เมื่อบอร์ดทำงานใน Program Mode
- **หมายเลข 5** คือ LED สีเขียว แสดงสถานะ RUN เมื่อบอร์ดทำงานใน Run Mode
- **หมายเลข 6** คือ Jumper สำหรับเลือกสัญญาณ ICSP เป็น PGEC2 และ PGED2
- **หมายเลข 7** คือ Jumper สำหรับเลือกสัญญาณ ICSP เป็น PGEC3 และ PGED3 ในกรณีเลือกใช้ MCU ตระกูล PIC32 ในกลุ่มที่ไม่มี USB(PIC32MX1XX)
- **หมายเลข 8** คือ Jumper สำหรับเลือกสัญญาณ ICSP เป็น PGEC1 และ PGED1
- **หมายเลข 9** คือ คือ Jumper สำหรับเลือกสัญญาณ ICSP เป็น PGEC3 และ PGED3 ในกรณีเลือกใช้ MCU ตระกูล PIC32 ในกลุ่มที่มี USB(PIC32MX2XX)
- **หมายเลข 10** คือ สวิตช์ Reset สำหรับสั่ง Reset การทำงานของบอร์ดเมื่อ MCU อยู่ในโหมด Run
- **หมายเลข 11** คือ ขั้วต่อสัญญาณ RA[0..4]
- **หมายเลข 12** คือ Crystal ค่าความถี่ 8.00MHz
- **หมายเลข 13** คือ Jumper สำหรับเลือกใช้สัญญาณนาฬิกาจาก Crystal หรือใช้จากวงจร FRC ภายในตัว MCU เอง ซึ่งสามารถใช้ RA2 และ RA3 ใช้งานเป็น GPIO ได้เมื่อเลือกใช้สัญญาณนาฬิกาจาก FRC ภายใน
- **หมายเลข 14** คือ LED แสดงสถานะของแหล่งจ่ายไฟ +5V ที่ป้อนให้บอร์ด
- **หมายเลข 15** คือ MCU ประจำบอร์ด เบอร์ PIC32MX250F128B
- **หมายเลข 16** คือ Jumper สำหรับเลือกติดตั้ง MCU ตระกูล PIC32MX ระหว่างกลุ่มที่มี USB และไม่มี โดย PIC32MX1XX(ไม่มี USB) และ PIC32MX2XX(มี USB)
- **หมายเลข 17** คือ Jumper สำหรับเลือกสัญญาณ RB6, RB10และ RB11 ว่าจะใช้เป็น GPIO หรือใช้เป็น USB
- **หมายเลข 18** คือ ขั้วต่อ USB แบบ HOST
- **หมายเลข 19** คือขั้วต่อ IDE10Pin ของ RB[0..7]
- **หมายเลข 20** คือขั้วต่อ IDE10Pin ของ RB[8..15]

- **หมายเลข 21** คือ Jumper สำหรับเลือก ON/OFF LED1 เพื่อเชื่อมต่อกับ RB14
- **หมายเลข 22** คือ LED1 สีเขียวสำหรับใช้แสดงสถานะสัญญาณที่ควบคุมโดย RB14
- **หมายเลข 23** คือ Jumper สำหรับเลือก ON/OFF LED2 เพื่อเชื่อมต่อกับ RB15
- **หมายเลข 24** คือ LED2 สีเขียวสำหรับใช้แสดงสถานะสัญญาณที่ควบคุมโดย RB15
- **หมายเลข 25** คือ Jumper สำหรับเลือกขาสัญญาณ RA0,RA4 ของ MCU ว่าจะให้เชื่อมต่อไปเป็น GPIOของ RA0 กับ RA4 หรือใช้เป็น TX1,RX1 ของ UART1
- **หมายเลข 26** คือ ขั้วต่อ UART1 โดยเป็นสัญญาณแบบ RS232 ซึ่งใช้ Pin ของ RA0(TX1) และ RA4(RX1) เป็นสัญญาณเชื่อมต่อ
- **หมายเลข 27** คือ Jumper สำหรับเลือกขาสัญญาณ RB8,RB9 ของ MCU ว่าจะให้เชื่อมต่อไปเป็น GPIOของ RB8 กับ RB9 หรือใช้เป็น TX2,RX2 ของ UART2
- **หมายเลข 28** คือ ขั้วต่อ UART2 โดยเป็นสัญญาณแบบ RS232 ซึ่งใช้ Pin ของ RB9(TX2) และ RB8(RX2) เป็นสัญญาณเชื่อมต่อ

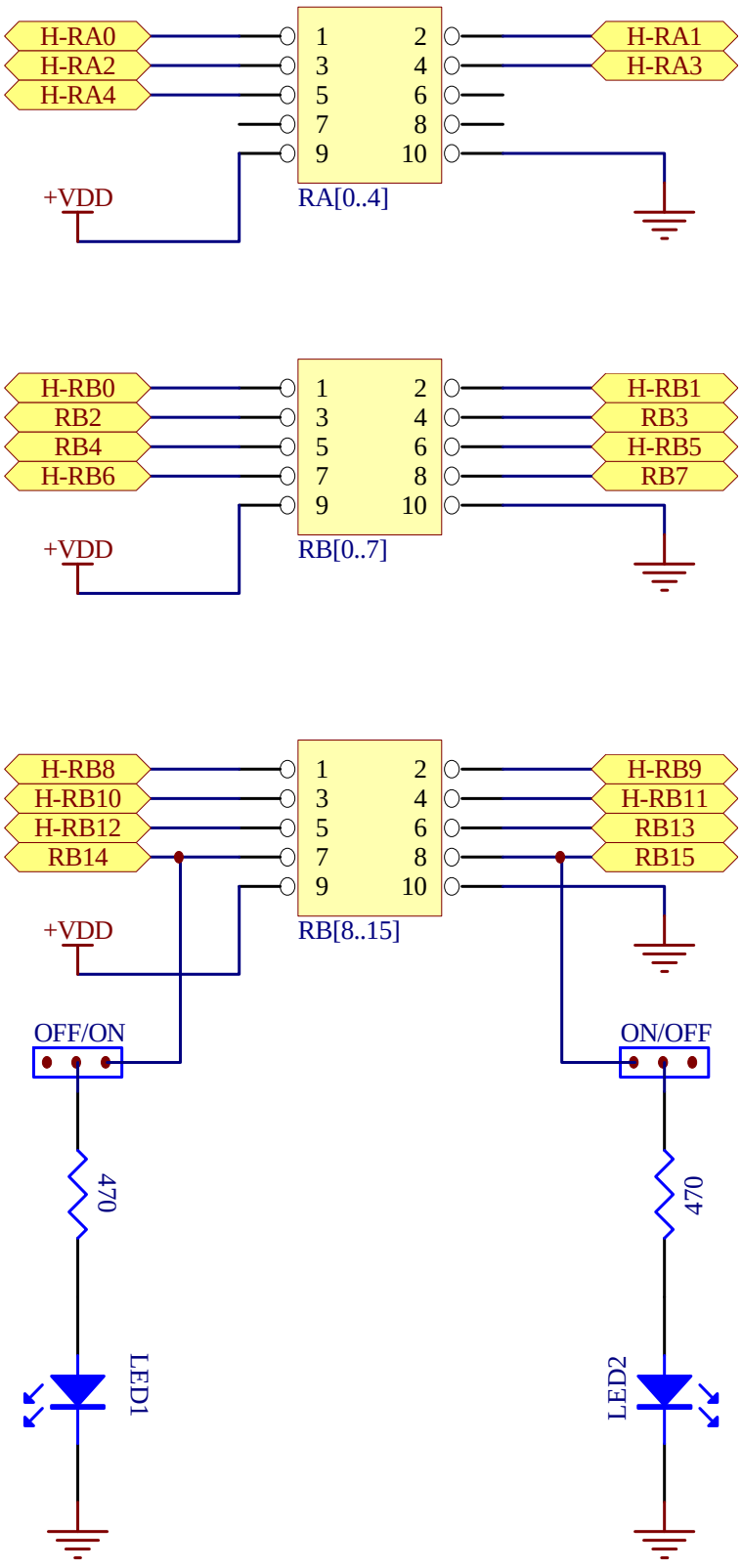
การจัดสรร I/O ของบอร์ด

PIC32MX1	PIC32MX2	Hardware ET-BASE PIC32
RA0/TX1	RA0/PGED3/TX1	RA0/PGED3/UART1
RA1	RA1/PGEC3	RA1/PGEC3
RA2/CKI	RA2/CKI	RA2/External Crystal
RA3/CKO	RA3/CKO	RA3/External Crystal
RA4/RX1	RA4/RX1	RA4/UART1
RB0/PGED1	RB0/PGED1	RB0/PGED1
RB1/PGEC1	RB1/PGEC1	RB1/PGEC1
RB2	RB2	RB2
RB3	RB3	RB3
RB4	RB4	RB4
RB5/PGED3	RB5	RB5/PGED3
RB6/PGEC3	RB6/VBUS	RB6/PGEC3/USB
RB7	RB7	RB7
RB8/RX2	RB8/RX2	RB8/UART2
RB9/TX2	RB9/TX2	RB9/UART2
RB10/PGED2	RB10/PGED2/USB D(-)	RB10/PGED2/USB
RB11/PGEC2	RB11/PGEC2/USB D(+)	RB11/PGED2/USB
RB12(PIC32MX1)	+VUSB3V3(PIC32MX2)	RB12/USB
RB13	RB13	RB13
RB14/LED1	RB14/LED1	RB14/LED1
RB15/LED2	RB15/LED2	RB15/LED2

*****หมายเหตุ*****

ขาสัญญาณ GPIO ของ PIC32MX สามารถเชื่อมต่อกับสัญญาณ Logic ที่มีขนาดแรงดันไม่เกิน 3.3V เท่านั้น ยกเว้น RB5, RB6, RB7, RB8, RB9, RB10 และ RB11 สามารถรับแรงดันของ Input Logic ที่เป็น 5Vได้

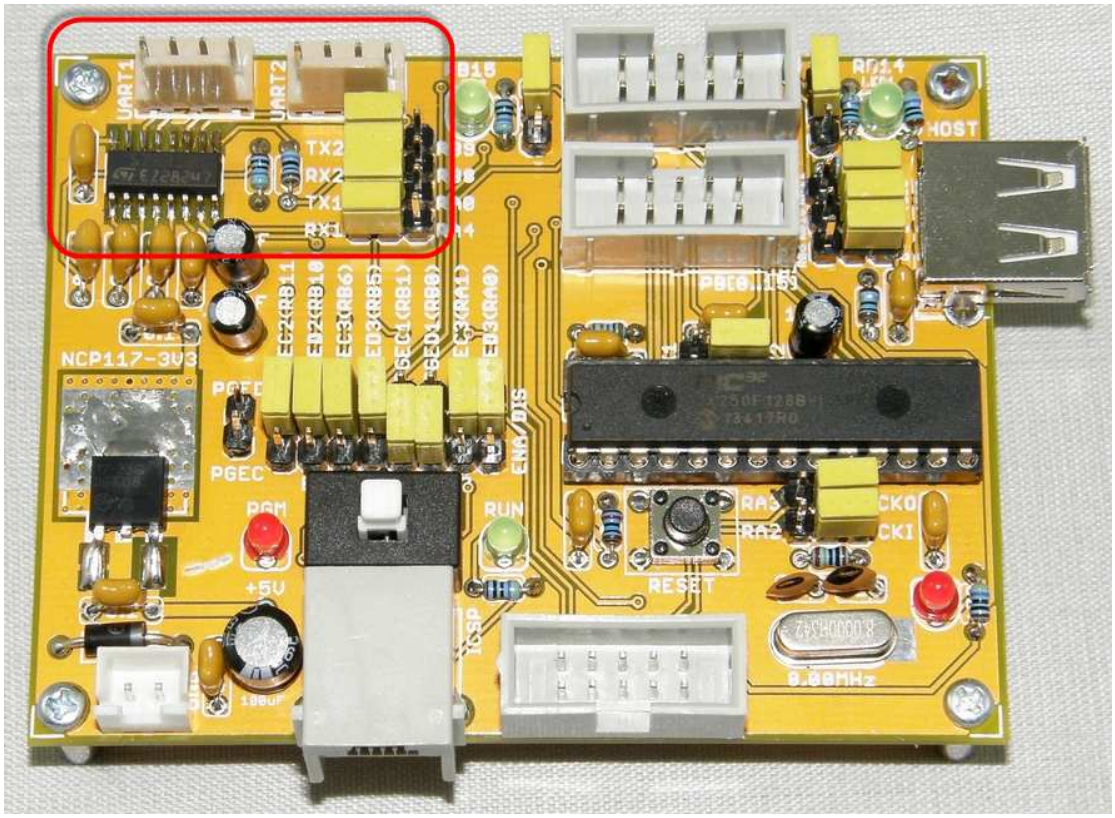
ขั้วต่อสัญญาณต่าง ๆ

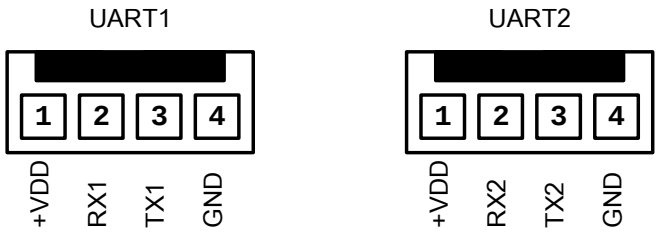


การจัดขั้ว GPIO ของบอร์ด

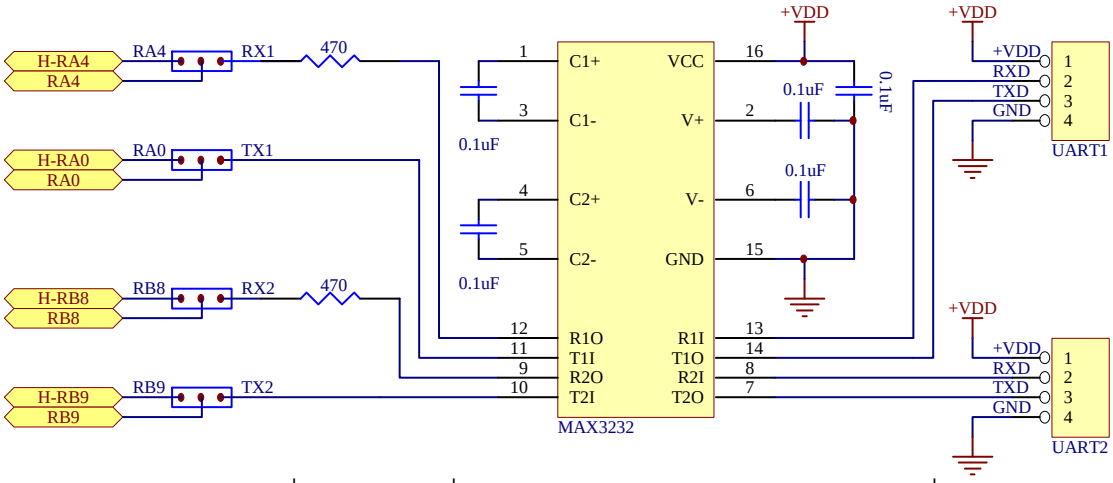
พอร์ตสื่อสารอนุกรม RS232

บอร์ด ET-BASE PIC32MX จะมีพอร์ตสื่อสารอนุกรมแบบ RS232 จำนวน 2พอร์ต โดยใช้ขั้วต่อแบบ CPA ขนาด 4 Pin เป็นจุดเชื่อมต่อสัญญาณที่ผ่านการแปลงระดับสัญญาณ TTL จาก Pin ของ MCU เป็นสัญญาณแบบ RS232 แล้ว ซึ่งสามารถใช้สื่อสารกับอุปกรณ์ต่างๆที่มีสัญญาณ RS232 มาตรฐานได้ทันที

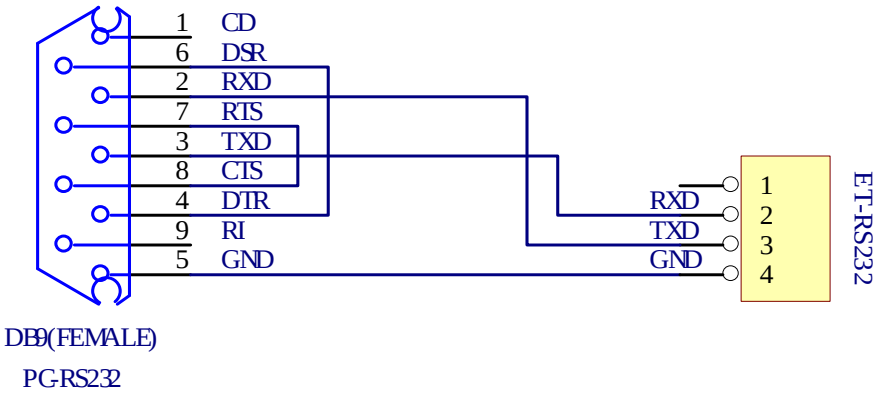




โดยวงจรการทำงานของ UART (RS232) ทั้ง 2 ช่อง สามารถเลือกใช้งาน หรือ ไม่ใช้งาน จาก Jumper ได้ เพื่อใช้เลือกว่าจะให้สัญญาณของ MCU ทำหน้าที่เป็น I/O หรือ UART ดังวงจร



สำหรับ Cable ที่จะใช้ในการเชื่อมต่อ RS232 ระหว่าง Comport ของเครื่องคอมพิวเตอร์ PC เข้ากับขั้วต่อ RS232 ของบอร์ด ET-BASE PIC32 นั้น เป็นดังนี้



รูป แสดงวงจรสาย Cable สำหรับ RS232

เนื่องจากระบบ UART ของ PIC32MX จะไม่ได้กำหนดขาสัญญาณในการเชื่อมต่อคงที่ตายตัวไว้เหมือนใน MCU บางเบอร์ แต่สามารถกำหนดจากคำสั่งในโปรแกรมเพื่อระบุการเชื่อมต่อ UART กับ Pin GPIO ของ MCU ได้เอง ซึ่งในบอร์ด ET-BASE PIC32MX จะกำหนดดังนี้

UART1

- RX1 จะใช้ Pin RA4 เป็นจุดเชื่อมต่อสัญญาณ
- TX1 จะใช้ Pin RA0 เป็นจุดเชื่อมต่อสัญญาณ

```
//Mapping RA4=RX1,RA0=TX1
PPSInput(3,U1RX,RPA4);        // Assign RPA4 as input pin for U1RX
PPSOutput(1,RPA0,U1TX);       // Set RPA0 pin as output for U1TX
```

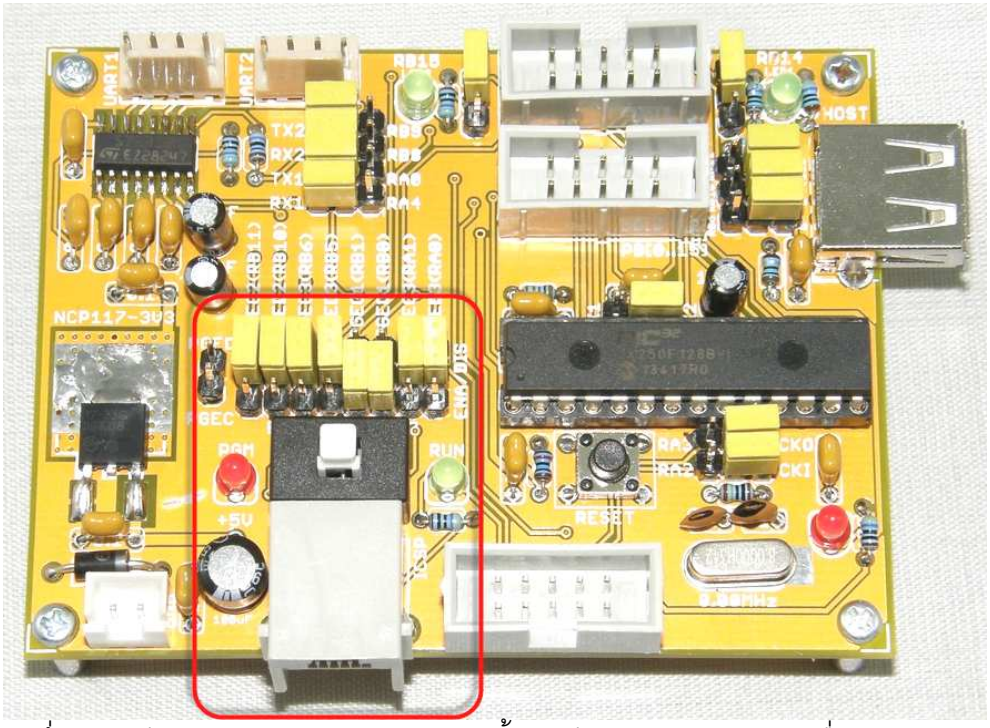
UART2

- RX2 จะใช้ Pin RB8 เป็นจุดเชื่อมต่อสัญญาณ
- TX2 จะใช้ Pin RB9 เป็นจุดเชื่อมต่อสัญญาณ

```
//Mapping RB8=RX2,RB9=TX2
PPSInput(2,U2RX,RPB8);        // Assign RPB8 as input pin for U2RX
PPSOutput(4,RPB9,U2TX);       // Set RPB9 pin as output for U2TX
```

การใช้งาน ICSP

ICSP จะเป็น Connector แบบ RJ11 สำหรับ Interface กับเครื่องมือพัฒนาโปรแกรมตระกูล PIC ที่มีการจัดขั้วตามมาตรฐาน ICSP ของ Microchips ที่รองรับการใช้งานร่วมกับ PIC32MX เช่น ICD3 หรือ Pickit3 ซึ่งสามารถใช้งานได้กับเครื่องมือพัฒนาของ Microchips หรือเทียบเท่า โดยจะมีสวิตช์สำหรับเลือกตัดต่อสัญญาณของ ICSP คือ PGEC, PGED และ MCLR สำหรับใช้ทำหน้าที่เชื่อมต่อกับ Programmer/Debugger หรือ ใช้งานตามปรกติได้ พร้อม LED แสดงสถานะ ว่าการทำงานของสวิตช์อยู่ในตำแหน่งใด โดยถ้าเลือกสวิตช์ไว้ทางด้าน Programmer/Debugger จะเห็น LED สีแดงของ PGM ติดสว่างให้เห็น แต่ถ้าตำแหน่งของสวิตช์อยู่ด้าน Run จะเห็น LED สีเขียว(RUN) ติดสว่างให้เห็น



ซึ่งในกรณีของ MCU ตระกูล PIC32MX นั้นจะมีขาสัญญาณสำหรับเชื่อมต่อกับ ICSP ให้เลือกใช้จำนวน 3 ชุด เพื่อให้ผู้ใช้สามารถเลือกสลับสับเปลี่ยนกันได้ในกรณีที่ขาสัญญาณ ICSP ไปซ้อนทับการทำงานกับฟังก์ชั่นพิเศษในตัว MCU ที่ผู้ใช้จำเป็นต้องใช้งาน โดยขาสัญญาณของ ICSP ใน PIC32MX สามารถทำหน้าที่เป็น ICSP สำหรับใช้โปรแกรม Flash Memory และใช้สำหรับทำการ Debug ในขาสัญญาณเดียวกัน โดยมีสัญญาณ 2 เส้น เรียกว่า PGEC และ PGED ซึ่งจะมีให้เลือกใช้ 3ชุด โดยถ้าจะใช้สำหรับเป็น ICSP ในการโปรแกรม Flash Memory สามารถเลือกใช้ชุดใดก็ได้ ทุกชุดสามารถทำหน้าที่ทดแทนกันได้หมด แต่สำหรับกรณีที่จะใช้ทำหน้าที่สำหรับ Debug ด้วย จะต้องกำหนดเลือกใน Configuration และสั่ง Enable การทำงานของ Debug ไว้ด้วย เช่น

ถ้าจะใช้เป็น ICSP ชุดที่1 ก็กำหนดค่าใน Configuration เป็น

```
#pragma config DEBUG = ON              // Debugger is Enabled
#pragma config ICESEL = ICS_PGx1       // ICSP = PGEC1/PGED1
```

ถ้าจะใช้เป็น ICSP ชุดที่2 ก็กำหนดค่าใน Configuration เป็น

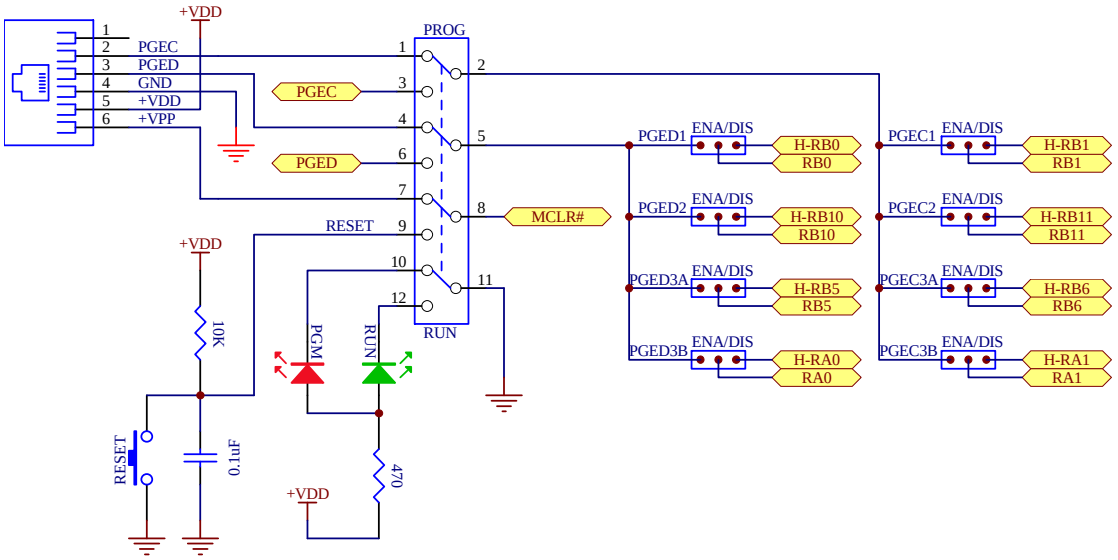
```
#pragma config DEBUG = ON              // Debugger is Enabled
#pragma config ICESEL = ICS_PGx2       // ICSP = PGEC2/PGED2
```

ถ้าจะใช้เป็น ICSP ชุดที่3 ก็กำหนดค่าใน Configuration เป็น

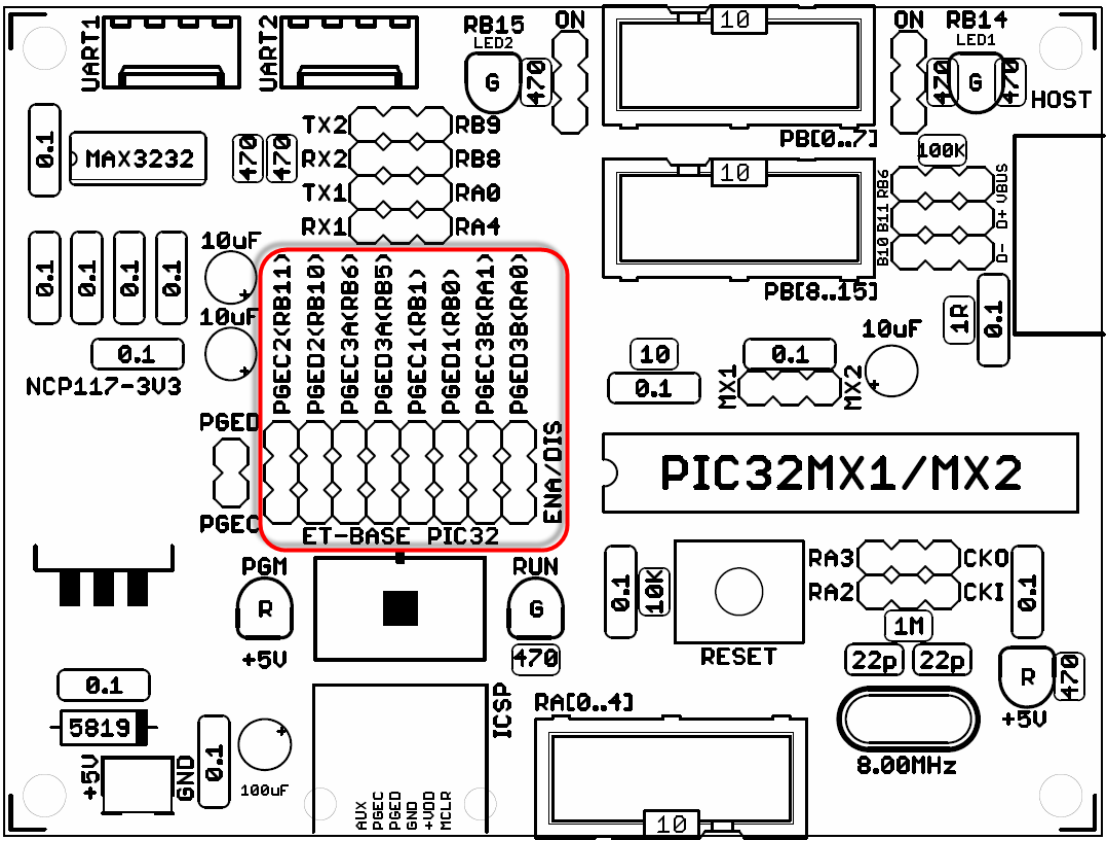
```
#pragma config DEBUG = ON              // Debugger is Enabled
#pragma config ICESEL = ICS_PGx3       // ICSP = PGEC3/PGED3
```

โดยขาสัญญาณของ ICSP ใน MCU ตระกูล PIC32MX มีการจัดวงจรและสัญญาณตามมาตรฐานของ ICSP ดังนี้

ICSP	PIC32MX1	PIC32MX2
PGED1	RB0	RB0
PGEC1	RB1	RB1
PGED2	RB10	RB10
PGEC2	RB11	RB11
PGED3	RB5	RA0
PGEC3	RB6	RA1



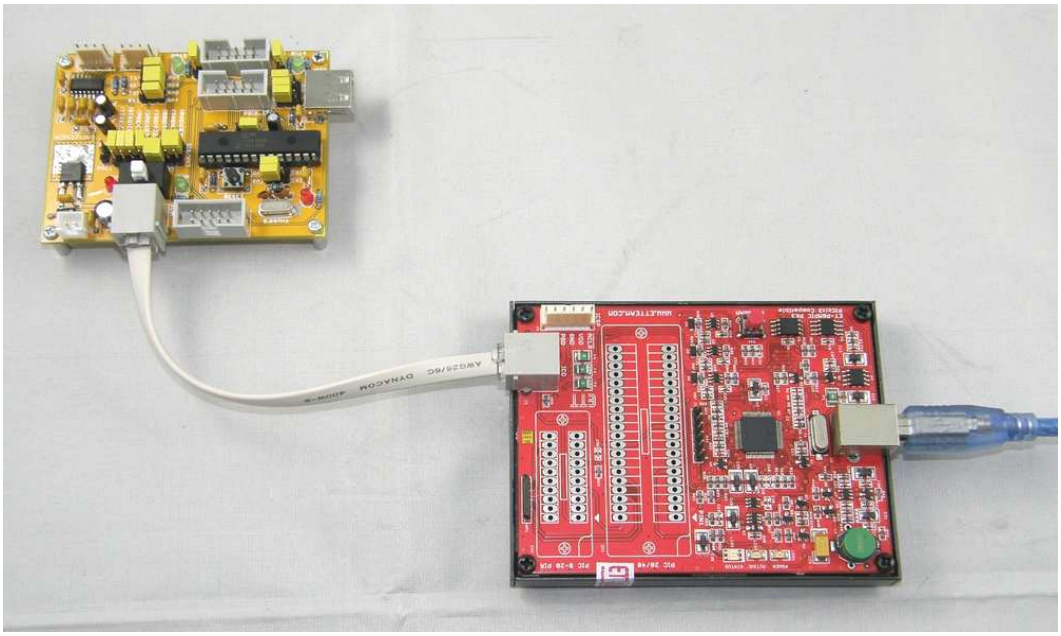
การจัดสัญญาณ ICSP/DEBUG ของบอร์ด



รูปแสดง ผังตำแหน่ง Jumper สำหรับเลือกใช้ ICSP ของ MCU ตระกูล PIC32MX

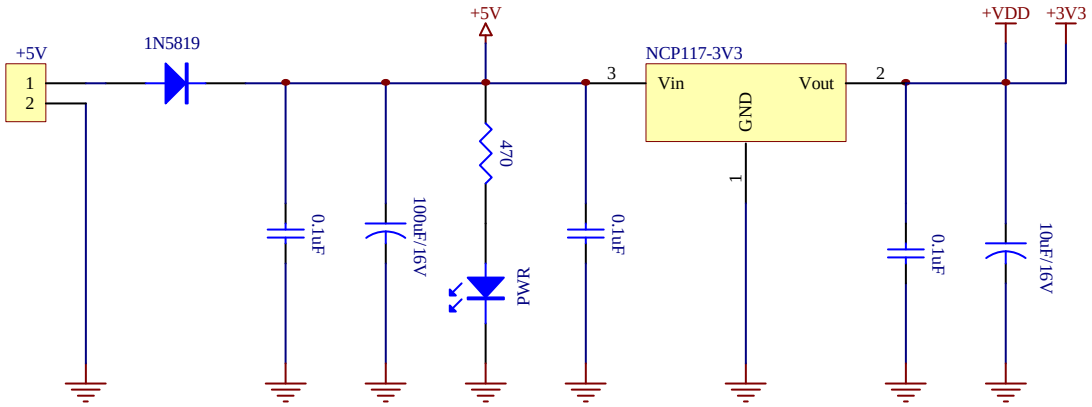
หมายเหตุ

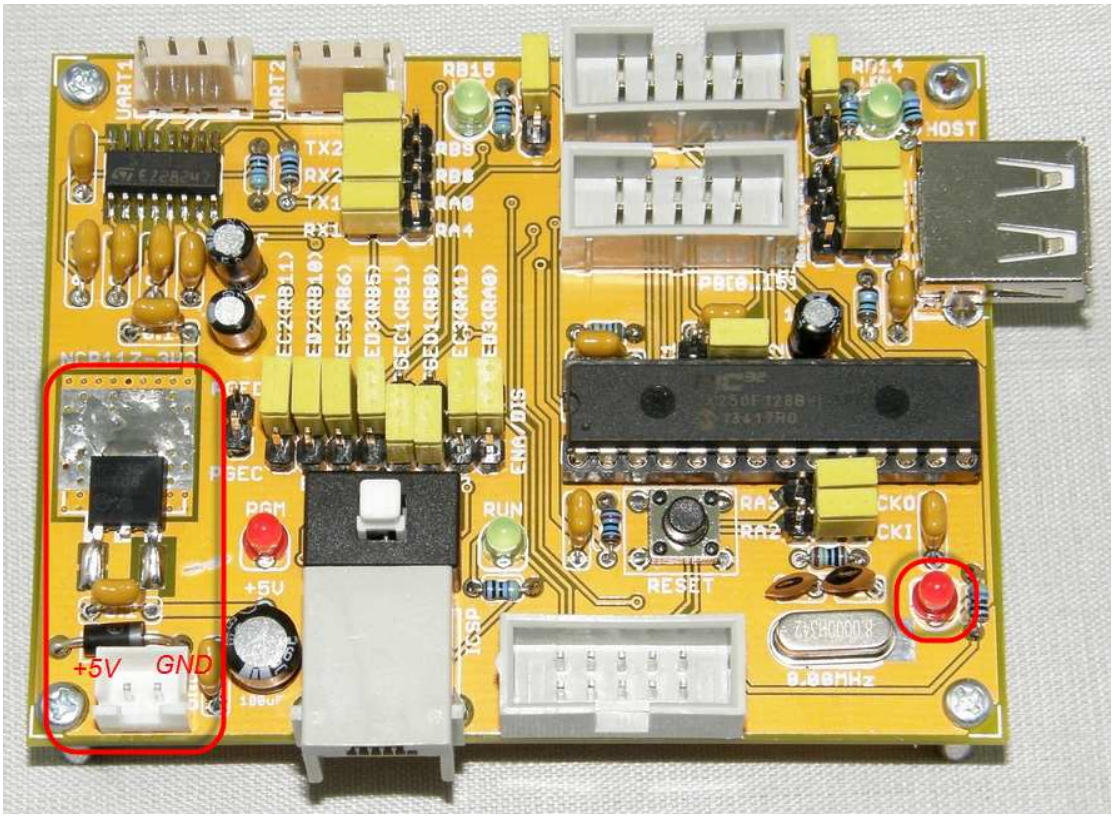
- PGEC3A และ PGED3A เป็นชุดขาสัญญาณ ICSP ชุดที่ 3 ของ PIC32MX1
- PGEC3B และ PGED3B เป็นชุดขาสัญญาณ ICSP ชุดที่ 3 ของ PIC32MX2



วงจรภาคจ่ายไฟ

สำหรับวงจรภาคจ่ายไฟของบอร์ด จะใช้วงจร Regulate ขนาด 3.3V/1A สามารถใช้งานได้กับไฟ DC ขนาด 5V/1A

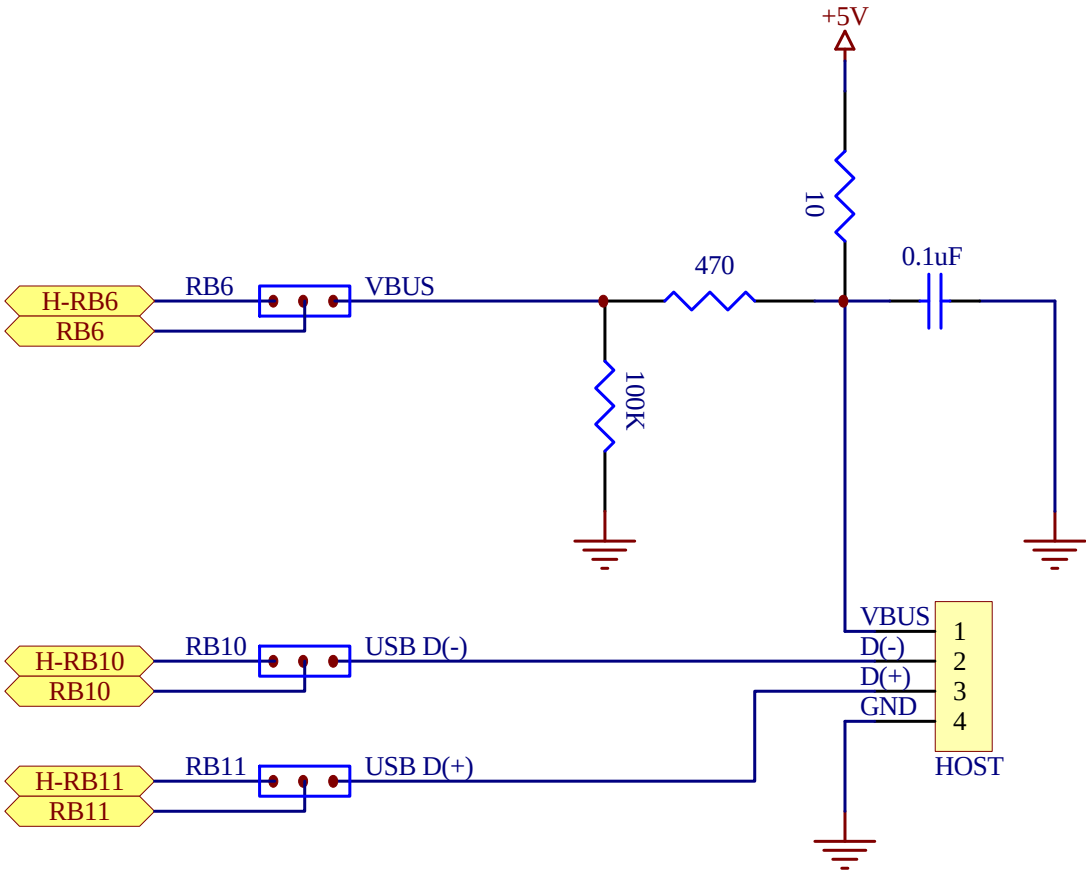


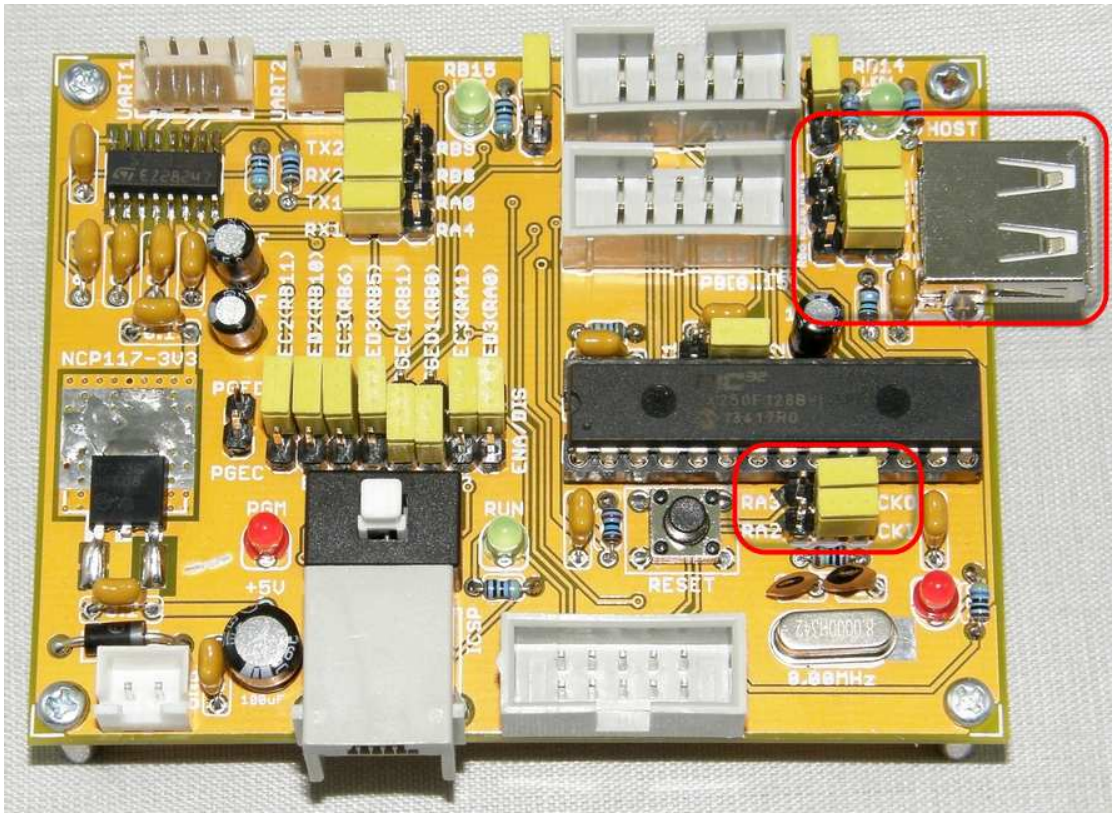


การใช้พอร์ต USB

ตามปรกติแล้ว PIC32MX2 จะมี Hardware ของ USB ซึ่งรองรับฟังก์ชันการทำงานทั้งแบบ Host (USB Host) และ Device (USB Device) และ OTG (USB ON-THE-GO) แต่สำหรับบอร์ด ET-BASE PIC32 นั้นจะออกแบบระบบ Hardware ให้รองรับเฉพาะ USB Host ไว้เท่านั้น โดยจะเสียขาสัญญาณไปทั้งหมด 6เส้น เพื่อใช้สำหรับติดต่อสื่อสารกับ USB 4 เส้น และใช้เป็นขาสัญญาณในการเชื่อต่อกับ Crystal Oscillator อีกจำนวน 2 เส้น เนื่องจากระบบ USB ต้องการสัญญาณนาฬิกาที่มีความแม่นยำสูง จึงไม่สามารถใช้กับสัญญาณนาฬิกาภายในจากวงจรกำเนิดความถี่ FRC 8MHz ได้ จำเป็นต้องใช้ขาสัญญาณ RA2 และ RA3 ทำหน้าที่เชื่อมต่อกับโมดูล Crystal ค่า 8MHz จากภายนอกตัว MCU แทน

- RB6 ทำหน้าที่เป็นขาตรวจจับแรงดันของ USB Bus Power(+VBUS)
- RB10 ทำหน้าที่เป็น USB D(-)
- RB11 ทำหน้าที่เป็น USB D(+)
- RB12 จะทำหน้าที่เป็นขารับแรงดันแหล่งจ่ายของ USB (VUSB3V3)





รูปแสดงตำแหน่งการเลือก Jumper เมื่อต้องการใช้งาน USB

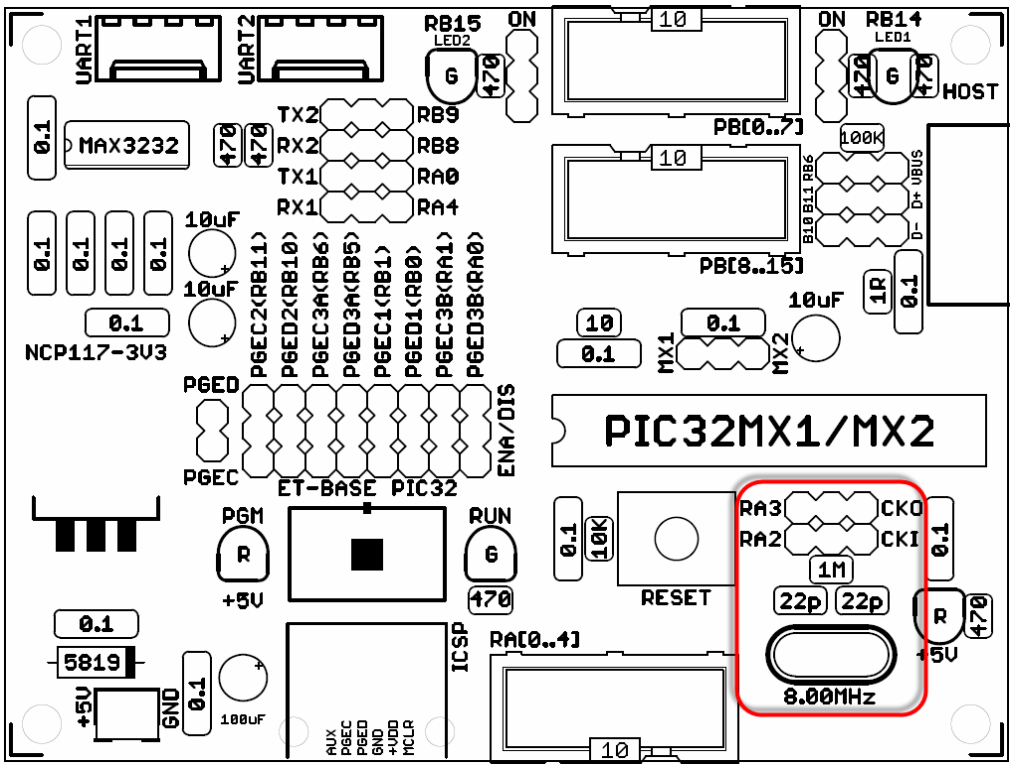
ระบบสัญญาณนาฬิกา

ระบบสัญญาณนาฬิกา Clock ของบอร์ด ET-BASE PIC32 สามารถเลือกได้ 2 แหล่ง คือ จาก Crystal ค่า 8MHz ภายนอก และจากวงจรกำเนิดภายใน FRC ค่า 8MHz โดยถ้าต้องการใช้สัญญาณนาฬิกาจาก Crystal ภายนอก จะต้องเสียขาสัญญาณ 2 เส้น คือ RA2 กับ RA3 สำหรับเชื่อมต่อกับ Crystal แต่ถ้าต้องการเลือกใช้สัญญาณนาฬิกาจากวงจรกำเนิดสัญญาณนาฬิกาภายใน ก็สามารถนำขาสัญญาณ RA2 และ RA3 ไปใช้งานเป็น GPIO หรือ ฟังก์ชั่นอื่นๆได้ตามต้องการ นอกจากนี้แล้วสัญญาณนาฬิกาของ PIC32MX ยังมีวงจรสำหรับใช้คูณค่าความถี่แบบ Phase Lock Loop(PLL) เพื่อทำการคูณความถี่จากสัญญาณนาฬิกาต้นกำเนิดให้มีค่าสูงขึ้นได้อีก โดย PIC32MAX สามารถ Run ได้สูงสุดที่ค่าความถี่ 50MHz

แต่อย่างไรก็ตาม ถ้าต้องมีการใช้งานฟังก์ชั่นเกี่ยวกับ USB ด้วย จำเป็นต้องใช้สัญญาณนาฬิกาจากวงจร Crystal ค่า 8MHz ภายนอกด้วยเสมอ เพราะระบบ USB มีความจำเป็นต้องใช้สัญญาณนาฬิกาที่มีความแม่นยำสูง ซึ่งวงจรกำเนิดสัญญาณนาฬิกาภายใน (FRC) มีความคลาดเคลื่อนที่ประมาณ 0.9% ไม่สามารถใช้กับ USB ได้

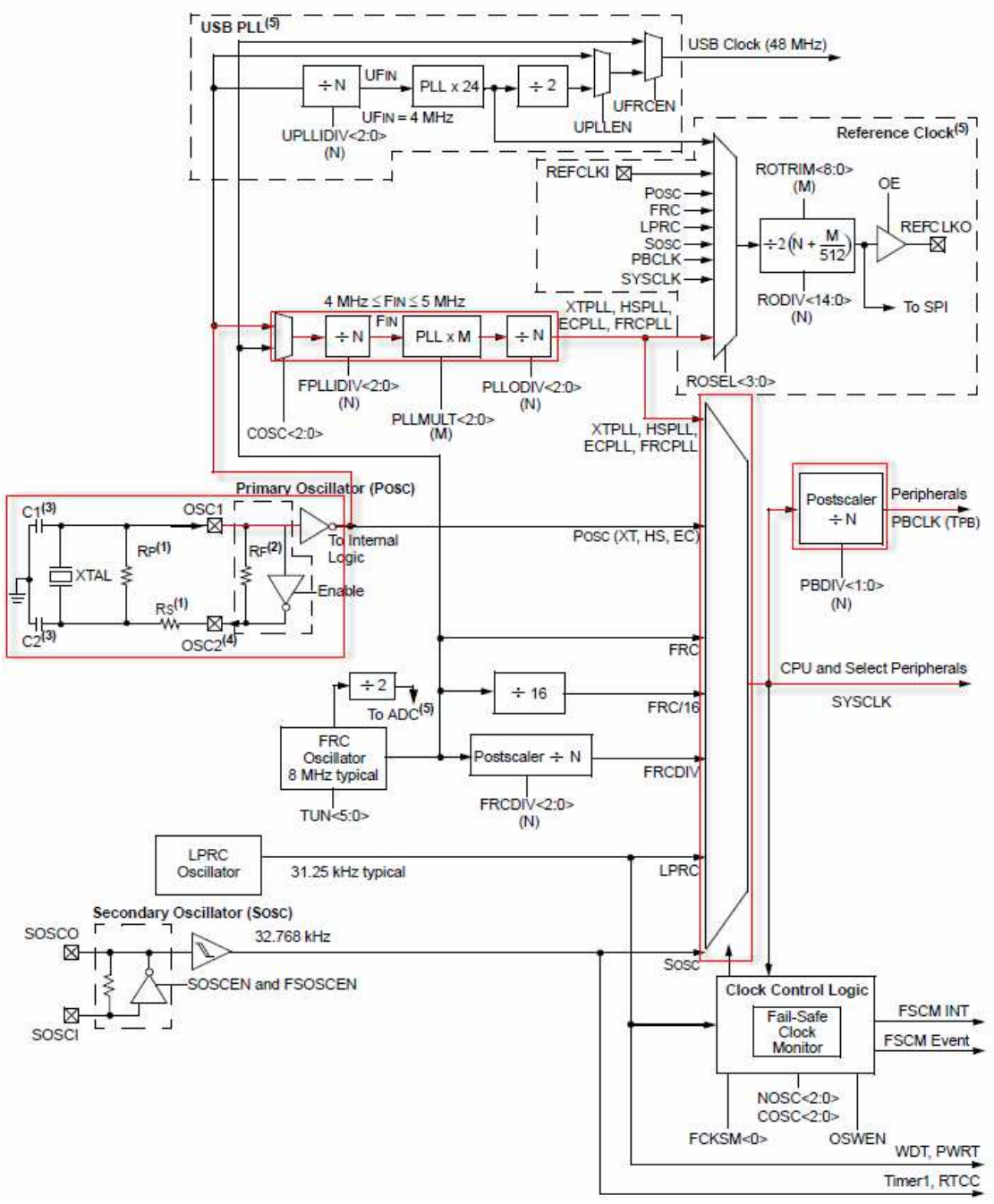
โดยในการเลือกใช้สัญญาณนาฬิกานั้น จะเลือกกำหนดผ่าน Configuration Bit ภายในตัว MCU เอง โดยทางด้านระบบ Hardware ของบอร์ดนั้นจะใช้ Jumper เป็นตัวเลือก โดยใช้

- RA2 ทำหน้าที่เป็น CKI สำหรับเชื่อมต่อกับ Crystal ภายนอก
- RA3 ทำหน้าที่เป็น CKO สำหรับเชื่อมต่อกับ Crystal ภายนอก



ตัวอย่างการใช้งานสัญญาณนาฬิกาจากแหล่งกำเนิดสัญญาณ Crystal ซึ่งมีค่าเป็น 8MHz ร่วมกับวงจรคูณความถี่ PLL เพื่อกำหนดค่าความถี่ให้เป็น 48MHz ซึ่งต้องเลือกกำหนดค่าใน Configuration ให้เลือกแหล่งกำเนิดสัญญาณนาฬิกาเป็น PRIPLL และคูณค่าความถี่เพื่อให้ได้ค่าความถี่ SYSCLK ที่ 48MHz ดังตัวอย่าง

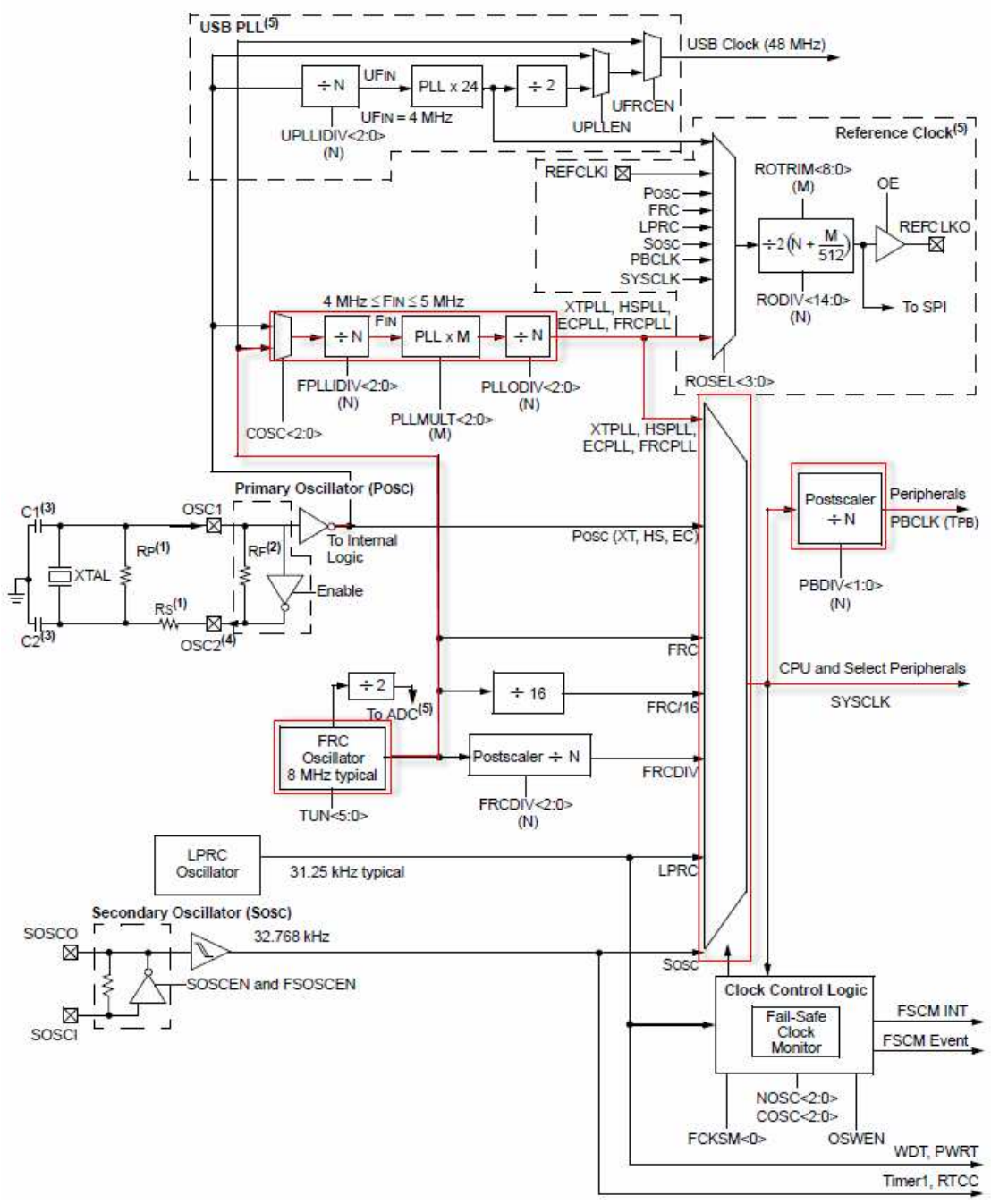
- กำหนดค่าหารความถี่ก่อนป้อนให้วงจร PLL ให้หาร 2
- กำหนดค่าอัตราการคูณความถี่ของวงจร PLL ให้คูณ 24
- กำหนดอัตราการหารความถี่หลังจากคูณความถี่แล้วของ PLL ให้หาร 2



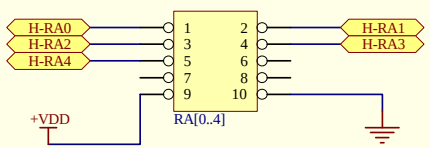
```
/*************************************************************/
/* Start of Config:PIC32MX1XX/2XX                            */
/* define Config Bit Name  in File : "pic32mx250f128b.h"     */
/* "\Microchip\xc32\v1.31\pic32-ibs\include\proc\p32mx..h"    */
/* define Config Bit Value in File : "32mx250f128b.html"     */
/* "\Microchip\xc32\v1.31\docs\config_docs\32mx250f128b.html" */
/*************************************************************/
// Configuration Bit settings
// Primary Osc = w / PLL (XT+,HS+,EC+PLL)
//             = POSCMOD : HS
//             = FNOSC   : PRIPLL
// SYSCLK      = 8MHz / FPLLIDIV(2) * FPLLMUL(24) / FPLLODIV(2)
//             = 8.0 MHz      / 2           * 24          / 2
//             = 48.0 MHz
// PBCLK       = SYSCLK(48)   / FPBDIV(1)
//             = 48.0MHz / 1
//             = 48.0 MHz
// WDT OFF

// DEVCFG[0]
#pragma config DEBUG = ON           // Debugger is Enabled
#pragma config JTAGEN = OFF         // JTAG Disabled
#pragma config ICESEL = ICS_PGx1    // ICSP = PGEC1/PGED1
#pragma config PWP = OFF            // Flash Write Protect = Disable
#pragma config BWP = OFF            // Boot Flash Write Protect = Disabled
#pragma config CP = OFF             // Code Protect = Disabled

// DEVCFG[1]
#pragma config FNOSC = PRIPLL   // Primary Osc w/PLL(XT+,HS+,EC+PLL)
#pragma config POSCMOD = HS      // Primary OSC = HS OSC Mode
#pragma config OSCIOFNC = ON        // CLKO Output on the OSCO = Enabled

#pragma config FSOSCEN = OFF        // Secondary Oscillator = Disabled
#pragma config IESO = OFF           // Internal/External SwitchOver Disabled
#pragma config FPBDIV = DIV_1       // PBCLK = SYSCLK / 1
#pragma config FCKSM = CSDCMD       // Clock Switch Disable, FSCM Disabled
#pragma config WDTPS = PS1048576    // Watchdog Timer Postscaler = 1:1048576
#pragma config WINDIS = OFF         // Watchdog Timer is in Non-Window Mode
#pragma config FWDTEN = OFF         // WDT Disabled (SWDTEN Bit Controls)
#pragma config FWDTWINSZ = WINSZ_75 // Watchdog Timer Window Size is 75%

// DEVCFG[2]
#pragma config FPLLIDIV = DIV_2     // PLL Input Divider = Div 2
#pragma config FPLLMUL = MUL_24     // Fin = PLL x 24
#pragma config UPLLIDIV = DIV_2     // Input USB PLL = Input Clock / 2
#pragma config UPLLEN = OFF         // USB PLL = Disabled and Bypassed
#pragma config FPLLODIV = DIV_2     // Output USB PLL = Output Clock / 2

// DEVCFG[3]
//#pragma config USERID = OFF       // USB USID = Controlled by Port
Function
#pragma config PMDL1WAY = OFF       // Allow multiple reconfigurations
#pragma config IOL1WAY = OFF        // Allow multiple reconfigurations
#pragma config FUSBIDIO = OFF       // USB USID = Controlled by Port
Function
#pragma config FVBUSONIO = OFF      // USB VBUS ON Controlled by Port

#define SYS_FREQ (48000000L)        // Run 48.0 MHz
#define LONG_DELAY  (SYS_FREQ/100)

#define  GetSystemClock()       (48000000ul)
#define  GetPeripheralClock()   (GetSystemClock()/(1 << OSCCONbits.PBDIV))
#define  GetInstructionClock()  (GetSystemClock())
```

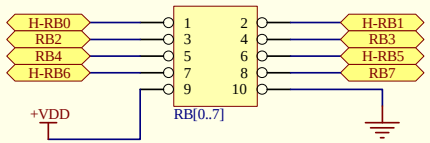
ตัวอย่างการใช้งานสัญญาณนาฬิกาจากแหล่งกำเนิดสัญญาณ FRC ภายในซึ่งมีค่าเป็น 8MHz ร่วมกับวงจรคูณความถี่ PLL เพื่อกำหนดค่าความถี่ให้เป็น 48MHz ซึ่งต้องเลือกกำหนดค่าใน Configuration ให้เลือกแหล่งกำเนิดสัญญาณนาฬิกาเป็น FRCPLL และคูณค่าความถี่เพื่อให้ได้ค่าความถี่ SYSCLK ที่ 48MHz ดังตัวอย่าง

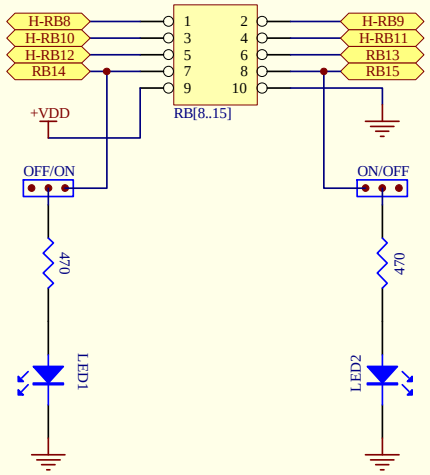
- กำหนดค่าหารความถี่ก่อนป้อนให้วงจร PLL ให้หาร 2
- กำหนดค่าอัตราการคูณความถี่ของวงจร PLL ให้คูณ 24
- กำหนดอัตราการหารความถี่หลังจากคูณความถี่แล้วของ PLL ให้หาร 2



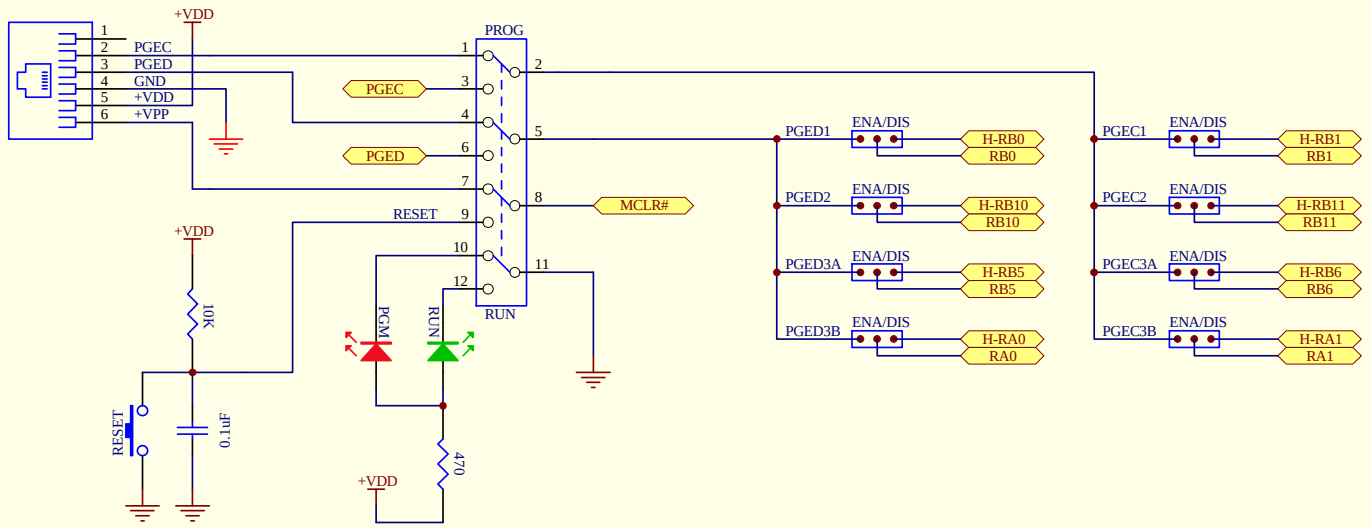
```
/*************************************************************/
/* Start of Config:PIC32MX1XX/2XX                            */
/* define Config Bit Name  in File : "pic32mx250f128b.h"     */
/* "\Microchip\xc32\v1.31\pic32-ibs\include\proc\p32mx..h"    */
/* define Config Bit Value in File : "32mx250f128b.html"     */
/* "\Microchip\xc32\v1.31\docs\config_docs\32mx250f128b.html" */
/*************************************************************/
// Configuration Bit settings
// Primary Osc = Fast RC Oscillator with PLL (FRCDIV+PLL)
//             = POSCMOD : OFF
//             = FNOSC   : FRCPLL
// SYSCLK      = 8MHz FRC / FPLLIDIV(2) * FPLLMUL(24) / FPLLODIV(2)
//             = 8.0 MHz  / 2           * 24          / 2
//             = 48.0 MHz
// PBCLK       = SYSCLK(48) / FPBDIV(1)
//             = 48.0MHz / 1
//             = 48.0 MHz
// WDT OFF

// DEVCFG[0]
#pragma config DEBUG = ON           // Debugger is Enabled
#pragma config JTAGEN = OFF         // JTAG Disabled
#pragma config ICESEL = ICS_PGx1    // ICSP = PGEC1/PGED1
#pragma config PWP = OFF            // Flash Write Protect = Disable
#pragma config BWP = OFF            // Boot Flash Write Protect = Disabled
#pragma config CP = OFF             // Code Protect = Disabled

// DEVCFG[1]
#pragma config FNOSC = FRCPLL       // Fast RC Oscillator + PLL
#pragma config POSCMOD = OFF        // Primary OSC = Disable
#pragma config OSCIOFNC = OFF       // CLKO Output on the OSCO = Disabled

#pragma config FSOSCEN = OFF        // Secondary Oscillator = Disabled
#pragma config IESO = OFF           // Internal/External SwitchOver Disabled
#pragma config FPBDIV = DIV_1       // PBCLK = SYSCLK / 1
#pragma config FCKSM = CSDCMD       // Clock Switch Disable, FSCM Disabled
#pragma config WDTPS = PS1048576    // Watchdog Timer Postscaler = 1:1048576
#pragma config WINDIS = OFF         // Watchdog Timer is in Non-Window Mode
#pragma config FWDTEN = OFF         // WDT Disabled (SWDTEN Bit Controls)
#pragma config FWDTWINSZ = WINSZ_75 // Watchdog Timer Window Size is 75%

// DEVCFG[2]
#pragma config FPLLIDIV = DIV_2     // PLL Input Divider = Div 2
#pragma config FPLLMUL = MUL_24     // Fin = PLL x 24
#pragma config UPLLIDIV = DIV_2     // Input USB PLL = Input Clock / 2
#pragma config UPLLEN = OFF         // USB PLL = Disabled and Bypassed
#pragma config FPLLODIV = DIV_2     // Output USB PLL = Output Clock / 2

// DEVCFG[3]
//#pragma config USERID = OFF       // USB USID = Controlled by Port
Function
#pragma config PMDL1WAY = OFF       // Allow multiple reconfigurations
#pragma config IOL1WAY = OFF        // Allow multiple reconfigurations
#pragma config FUSBIDIO = OFF       // USB USID = Controlled by Port
Function
#pragma config FVBUSONIO = OFF      // USB VBUS ON Controlled by Port

#define SYS_FREQ (48000000L)        // Run 48.0 MHz
#define LONG_DELAY  (SYS_FREQ/100)

#define  GetSystemClock()        (48000000ul)
#define  GetPeripheralClock()   (GetSystemClock()/(1 << OSCCONbits.PBDIV))
#define  GetInstructionClock()  (GetSystemClock())
```



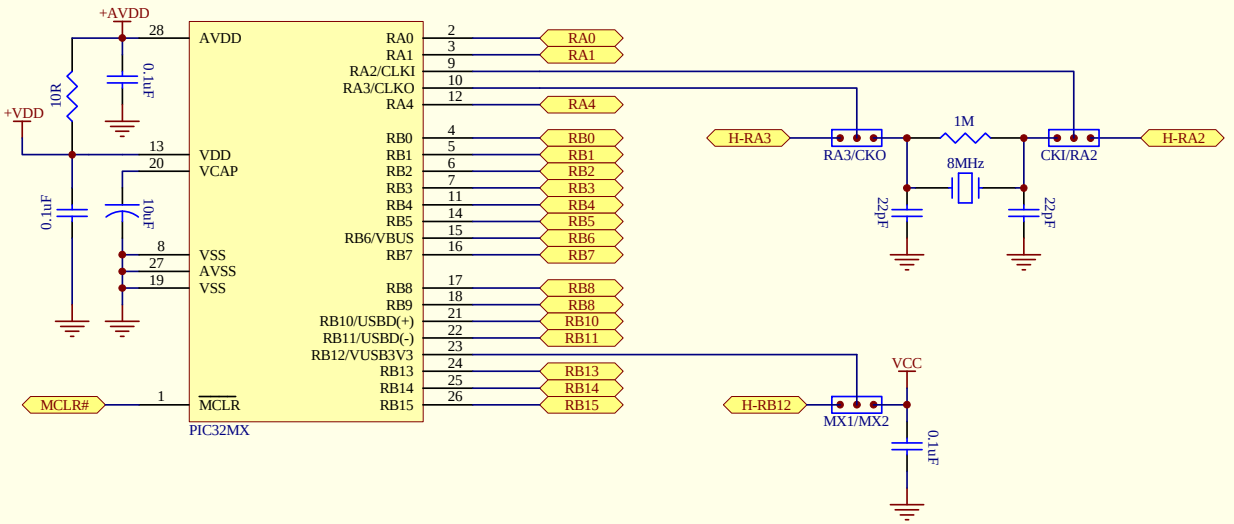




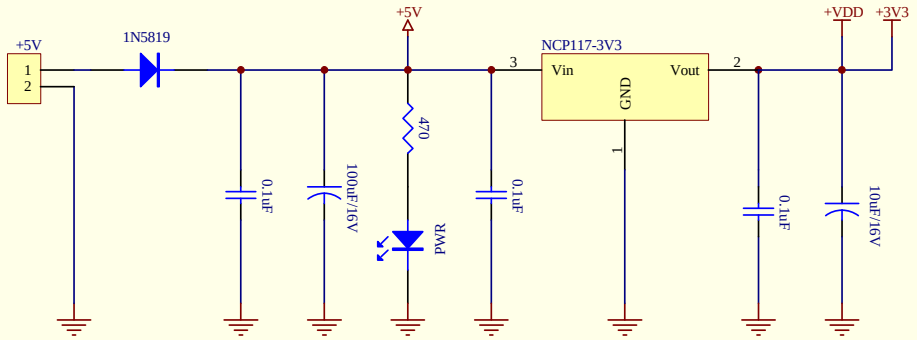
Title	ET-BASE PIC32MX250F128B (WWW.ETT.CO.TH)	
Size: A4	Number: MCU UNIT	Revision: 1.0
Date: 14-Jul-2014		Sheet 2 of 6
File: C:\sch\base-pic32\BASE-PIC32MX.Ddb		Drawn By:



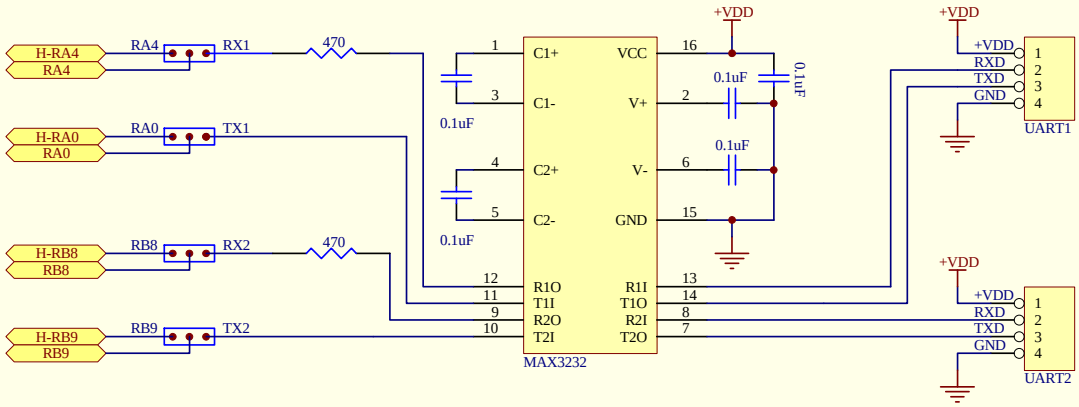
Title	ET-BASE PIC32MX250F128B (WWW.ETT.CO.TH)	
Size: A4	Number: MCU UNIT	Revision: 1.0
Date: 14-Jul-2014		Sheet 3 of 6
File: C:\sch\base-pic32\BASE-PIC32MX.Ddb		Drawn By:



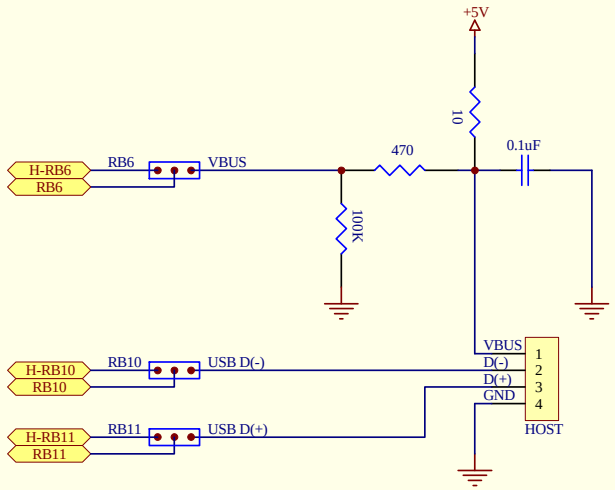
Title	ET-BASE PIC32MX250F128B (WWW.ETT.CO.TH)		
Size: A4	Number: MCU UNIT	Revision: 1.0	
Date: 14-Jul-2014		Sheet 1 of 6	
File: C:\sch\base-pic32\BASE-PIC32MX.Ddb		Drawn By:	



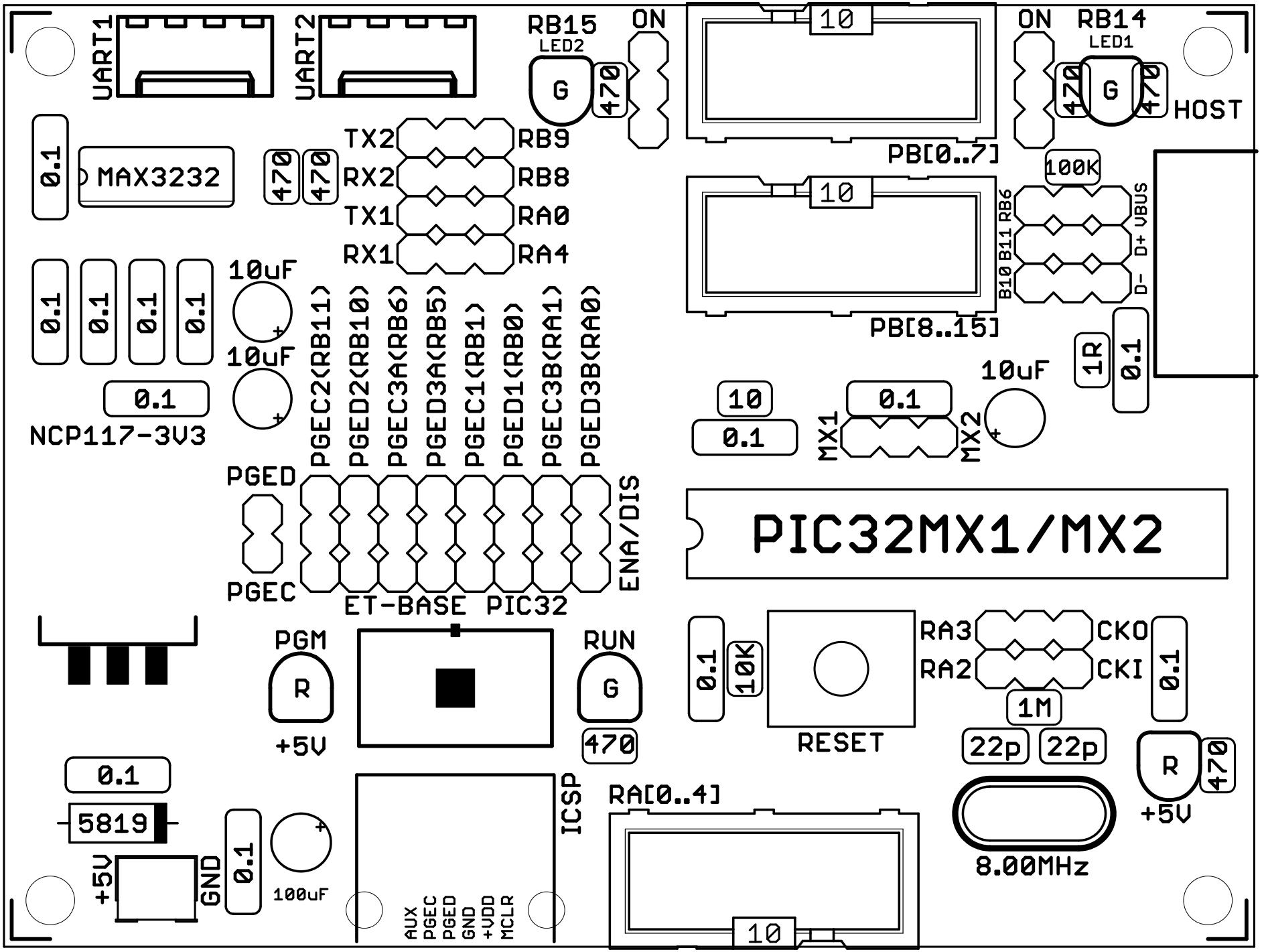
Title	ET-BASE PIC32MX250F128B (WWW.ETT.CO.TH)		
Size: A4	Number: MCU UNIT	Revision: 1.0	
Date: 15-Jul-2014		Sheet 4 of 6	
File: C:\sch\base-pic32\BASE-PIC32MX.Ddb		Drawn By:	



Title	ET-BASE PIC32MX250F128B (WWW.ETT.CO.TH)	
Size: A4	Number: MCU UNIT	Revision: 1.0
Date: 14-Jul-2014		Sheet 5 of 6
File: C:\sch\base-pic32\BASE-PIC32MX.Ddb		Drawn By:



Title	ET-BASE PIC32MX250F128B (WWW.ETT.CO.TH)		
Size: A4	Number: MCU UNIT	Revision: 1.0	
Date: 14-Jul-2014		Sheet 6 of 6	
File: C:\sch\base-pic32\BASE-PIC32MX.Ddb		Drawn By:	


UART1
UART2
RB15
LED2
G
470
ON
10
PB[0..7]
ON
RB14
LED1
470
G
470
HOST
0.1
MAX3232
470
470
TX2
RX2
TX1
RX1
RB9
RB8
RA0
RA4
10
PB[8..15]
100K
B10 B11 RB6
D- D+ VBUS
0.1
0.1
0.1
0.1
10uF
10uF
1R
0.1
0.1
NCP117-3V3
10
0.1
MX1
0.1
MX2
10uF
PGED2(RB11)
PGED2(RB10)
PGEC3A(RB6)
PGED3A(RB5)
PGEC1(RB1)
PGED1(RB0)
PGEC3B(RA1)
PGED3B(RA0)
PGED
PGEC
ENA/DIS
ET-BASE PIC32
PIC32MX1/MX2
PGM
R
+5V
RUN
G
470
0.1
10K
RESET
RA3
RA2
CKO
CKI
1M
22p
22p
0.1
R
470
+5V
0.1
5819
+5V
GND
0.1
100uF
ICSP
AUX
PGEC
PGED
GND
+VDD
MCLR
RA[0..4]
10
8.00MHz