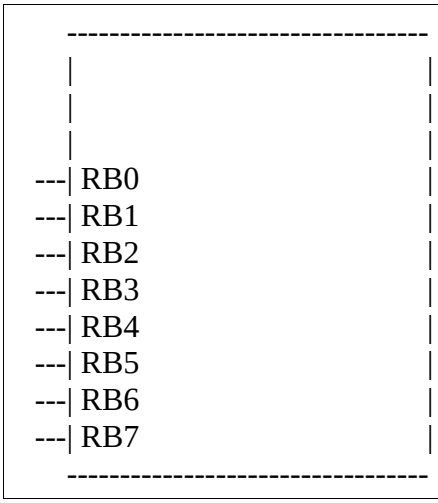
CPEN 305 – Reference Sheet

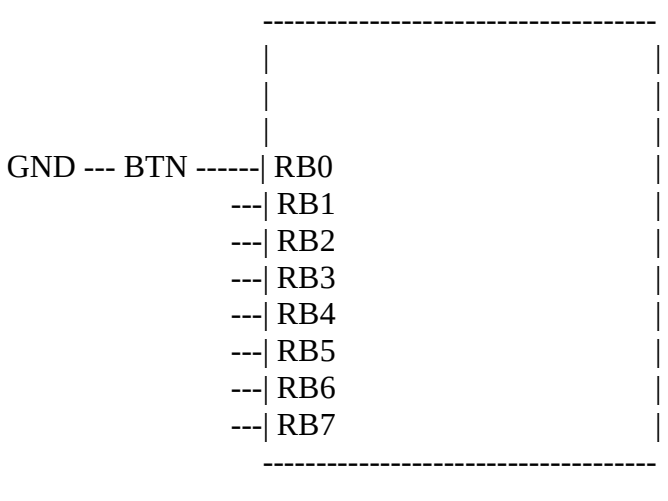
Dr. Edgard Chammas

Enable internal pull-ups on PORTB pins




```
INTCON2bits.RBPU = 0;
```

Classical (non-flag) way to handle push buttons

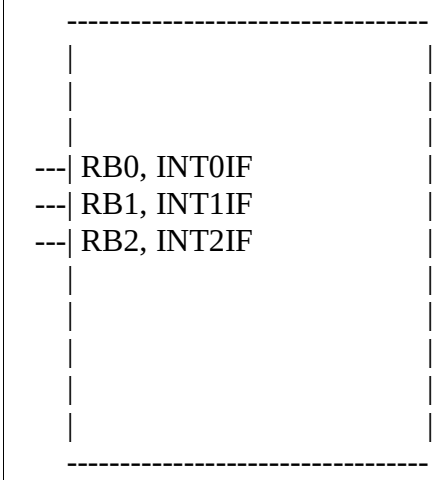



```
while(1) {
  if(PORTBbits.RB0) {
    ...
    while(PORTBbits.RB0);
  }
}

while(1) { //If BTN is pulled-up
  if(!PORTBbits.RB0) {
    ...
    while(!PORTBbits.RB0);
  }
}
```

INTxIF - Pin Edge Flags

Affected pins



Polling mode usage example

```
while(1) {
  if(INTCONbits.INT0IF) {
    INTCONbits.INT0IF = 0;
    ...
  }
}
```

Interrupt mode usage example

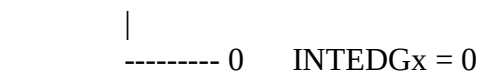
```
#pragma code ISR = 0x0008
#pragma interrupt ISR

void ISR(void)
{
  INTCONbits.INT0IF = 0;
  ...
}
```

Flags referencing

```
INT0IF   ===>   INTCONbits.INT0IF
INT1IF   ===>   INTCON3bits.INT1IF
INT2IF   ===>   INTCON3bits.INT2IF
```

Configure pin edge flags to react on falling edge




```
INTCON2bits.INTEDG0 = 0;
INTCON2bits.INTEDG1 = 0;
INTCON2bits.INTEDG2 = 0;
```

Configure pin edge flags to react on rising edge

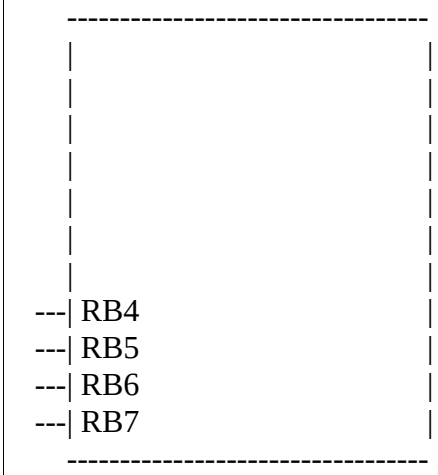

```
INTCON2bits.INTEDG0 = 1;
INTCON2bits.INTEDG1 = 1;
INTCON2bits.INTEDG2 = 1;
```

Configure pin edge flags to run in interrupt mode

```
INTCONbits.GIE = 1;
INTCONbits.INT0IE = 1;
INTCON3bits.INT1IE = 1;
INTCON3bits.INT2IE = 1;
```

RBIF - Pin Change Flag

Affected pins



Polling mode example

```
while(1) {
  if(INTCONbits.RBIF) {
    PORTB = PORTB;
    INTCONbits.RBIF = 0;
    ...
  }
}
```

Flag referencing

RBIF ===> INTCONbits.RBIF

Configure pin change flag to run in interrupt mode

```
INTCONbits.GIE = 1
INTCONbits.RBIE = 1
```

Interrupt mode usage example

```
#pragma code ISR = 0x0008
#pragma interrupt ISR

void ISR(void)
{
  PORTB = PORTB;
  INTCONbits.RBIF = 0;
  ...
}
```

Timer 0 Configuration register – T0CON

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	TOPS2	TOPS1	TOPS0

Bit 7	**TMR0ON: Timer0 On/Off Control bit** 1 = Enables Timer0 0 = Stops Timer0
Bit 6	**T08BIT: Timer0 8-Bit/16-Bit Control bit** 1 = Timer0 is configured as an 8-bit timer/counter 0 = Timer0 is configured as a 16-bit timer/counter
Bit 5	**T0CS: Timer0 Clock Source Select bit** 1 = Transition on T0CKI pin 0 = Internal instruction cycle clock (CLKOUT)
Bit 4	**T0SE: Timer0 Source Edge Select bit** 1 = Increment on high-to-low transition on T0CKI pin 0 = Increment on low-to-high transition on T0CKI pin
Bit 3	**PSA: Timer0 Prescaler Assignment bit** 1 = Timer0 prescaler is not assigned. Timer0 clock input bypasses prescaler. 0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.
Bit 2-0	**T0PS<2:0>: Timer0 Prescaler Select bits** 111 = 1:256 Prescale value 110 = 1:128 Prescale value 101 = 1:64 Prescale value 100 = 1:32 Prescale value 011 = 1:16 Prescale value 010 = 1:8 Prescale value 001 = 1:4 Prescale value 000 = 1:2 Prescale value

Polling mode usage example

```
while(1) {
  if(INTCONbits.TMR0IF) {
    INTCONbits.TMR0IF = 0;
    TMR0H = (65536 – n) / 256;
    TMR0L = (65536 – n) % 256;
    ...
  }
}
```

Interrupt mode usage example

```
#pragma code ISR = 0x0008
#pragma interrupt ISR

void ISR(void)
{
  INTCONbits.TMR0IF = 0;
  TMR0H = (65536 – n) / 256;
  TMR0L = (65536 – n) % 256;
  ...
}
```

Configure Timer 0 to run in interrupt mode

```
INTCONbits.GIE = 1;
INTCONbits.TMR0IE = 1;
```