PIC24FKA301 Tips & Tricks

Header File Macros:

- **Any register** found in the data sheet can be accessed directly by its name
  - ex. to set register PR1 (period for timer 1)
    - PR1 = 0x7FFF;
- **Individual bits in a register** can be accessed in the following way
  - TRISAbits.TRISA0 = 1;
- **Any named bit** within a register can also be accessed using its name preceded by an underscore, as defined in the header file
  - ex. to set T1IF (timer 1 interrupt flag) bit in IFS0 (interrupt flag register 0)
    - _T1IF = 1;
  - **EXCEPTION:** bit names for peripherals with multiple modules (timers, OC modules) only apply to module 1. Others must be written out explicitly
    - ex. T2Conbits.TCKPS = 0b10; instead of _TCKPS = 0b10;

Digital I/O:

- Be sure to **disable analog input** on any pins being used for digital I/O
  - ex. ANSA = 0; (for pins on port A)
- It is good practice to **write to latches** rather than directly to pins
  - ex. _LATA6 = 1; instead of _RA6 = 1;
  - Writes to pins should be interpreted by the compiler as writes to the latch, but there have been problems with this in the past
- **Pins 9 and 10** (RB4 and RA4) are shared with the secondary oscillator peripheral. To use these pins for digital I/O you need to clear the SOSCSRC bit of the FOSCSEL configuration register. This can be accomplished when you select your oscillator source, e.g., _FOSCSEL(FNOSC_LPRC & SOSCSRC_DIG);
- **Pin 8** (RA3) defaults as a clock output (CLKO). This functionality must be disabled in order to use pin 8 for digital I/O. Include the following code outside your main function:
  - _FOSC(OSCIOFNC_OFF)
- **Pin 1** can be used as a digital input (*not output*) if the MCLRE bit in the FPOR register is set to 0. Include the following code outside your main function:
  - _FPOR(MCLRE_OFF)

Interrupts:

- **A complete list of defined interrupt names** can be found at “C:\Program Files (x86)\Microchip\xc16\v1.21\support\PIC24F\gld\p24F16KA301.gld” beginning at line 254. Here are some of the most useful ones (replace ‘x’ with number of module you

use):
    - _INTxInterrupt
    - _OCxInterrupt
    - _TxInterrupt
    - _ADC1Interrupt
    - _CNInterrupt
    - _CompInterrupt
- To avoid **"PSV model not specified" error**, replace _ISR with the following
    - __attribute__((interrupt, no_auto_psv))
- For **input change notification** (CN) you must enable individual pin and general interrupt
    - ex. _CN5IE = 1;   // specific pin interrupt enable
    -         _CNIE = 1;   // global interrupt enable
- **Make sure there is a space** between _ISR and _T1Interrupt(void). The correct line should read "void _ISR _T1Interrupt(void)". Without a space it will build fine and give no errors or warning, but the interrupt will not work.
- **Do not make function calls inside interrupts.** It will build fine, but it will get stuck at the function. In general, interrupts are designed for very specific actions that do not involve making function calls. If necessary, include the lines from the function in the interrupt.
- **OC/PWM interrupts**
    - If you have selected a timer in OCTSEL, enable and use that timer's interrupt
    - If you are using the system clock, enable and use the OCx interrupt

Disassembly:

- **To view assembly code** for a c function in MPLABX
    - Build project
    - Window -> Output -> Disassembly Listing File
    - This will display a new window including your c code broken down by line with the associated assembly instructions underneath.

Defaults:

- Oscillator - 8MHz FRCDIV
- Oscillator Postscaler (_RCDIV) - divide by 2

A/D Conversion

- **To scan multiple channels** in auto convert mode:
    - _SSRC to specify auto conversion
    - _ASAM set for auto sampling

- AD1CSSL selects which channels to scan
- _BUFREGEN selects whether to use FIFO mode or channel indexed mode (AN1 stored in ADC1BUF1, AN2 stored in ADC1BUF2, etc.)
- _CSCNA set to scan channels specified in AD1CSSL instead of CH0SA bits
- _ALTS = 0 to sample MUXA only
- _SMPI = X to specify when interrupts occur. When sampling two channels, for example, interrupts should occur at every other sample.

MPLAB and PICkit

- You can manually **create a collapsible section of code** by typing:

  ```
  fcom + <Tab>
  ```
- To **change which set of programming pins** on the PIC are used insert the following outside your main function:

  _FICD(ICS_PGx1);

  Where you replace the '1' with the number of the pin pair you wish to use.