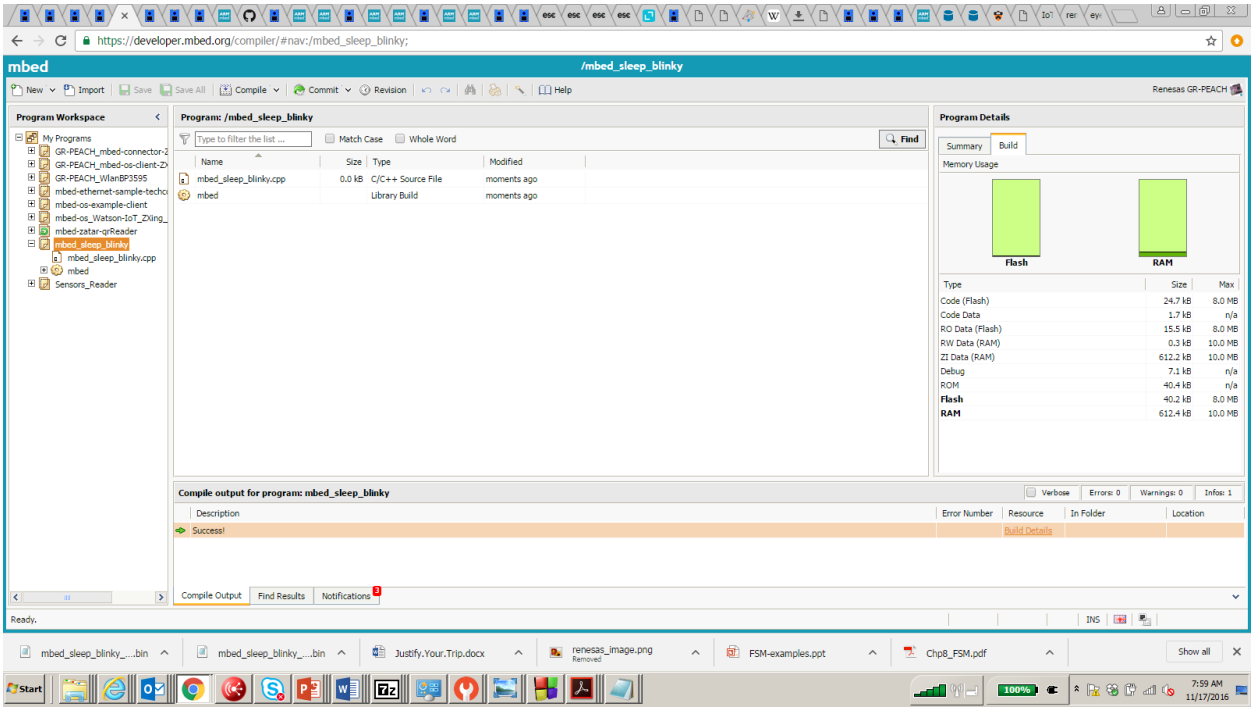
Test WFI (Wait for Interrupt) Sleep Blinky example on mbed5.X

https://developer.mbed.org/users/pateshian/code/mbed_sleep_blinky/

https://developer.mbed.org/compiler/#nav:/mbed_sleep_blinky;

```
#include "mbed.h"
#define IRQ4    (36)
gpio_irq_t irqHandler;
DigitalOut led1(LED1);
int gFlag = 0;
void interrupt_irq4_user (uint32_t id, gpio_irq_event event) {
   if (!gFlag){
      gFlag = 1;
   }
}
// main() runs in its own thread in the OS
// (note the calls to Thread::wait below for delays)
int main() {
      gpio_irq_init (&irqHandler, USER_BUTTON0, interrupt_irq4_user, IRQ4);
      while (true) {
      if ( gFlag == 1 ) {
         gFlag = 0;
         led1 = led1;
         __WFI();
         led1 = !led1;
      }
      //led1 = !led1;
      //Thread::wait(500);
   }
}
```

Compile success results:



Four methods to implement SLEEP:

> The generic code below accomplishes this now on GR-PEACH platform, Compiles SUCESSFULLY On mbed5.X Online compiler, Proven sleep and wake up using four different modes and meets all the given requirements.
>
> NOTE: There are also device specific ways to implement sleep which require device specific register settings to selectively power down caches, peripherals and other resources. These are covered in device specific application notes.

```
/*** Main function ---------------------------------------------------------- ***/
#define D_SLEEP_VERSION 4

/* Generic main function/loop for enabling WFI in case of
  Interrupt based cyclic execution
*/
   /* Start timer irq */
   ticker.attach_us(timer_irq, MS_INTERVALS * APP_LOOP_PERIOD);

#if D_SLEEP_VERSION == 1 // version without WFI/WFE and without IRQ synchronization
```

```
  while (true) {
    if(timer_irq_triggered) {
      timer_irq_triggered = false;
      main_cycle();
    } else if(ff_irq_triggered) {
      ff_irq_triggered = false;
      handle_ff_irq();
    }
  }
#elif D_SLEEP_VERSION == 2 // classical version with WFI and with IRQ
synchronization
  while (true) {
    __disable_irq();
    if(timer_irq_triggered) {
      timer_irq_triggered = false;
      __enable_irq();
      main_cycle();
    } else if(ff_irq_triggered) {
      ff_irq_triggered = false;
      __enable_irq();
      handle_ff_irq();
    } else {
      __WFI();
            __enable_irq(); /* do NOT enable irqs before WFI to avoid
          opening a window in which you can loose
          irq arrivals before going into WFI */
    }
  }
#elif D_SLEEP_VERSION == 3 // version with WFE and with IRQ synchronization
  while (true) {
    __disable_irq();
    if(timer_irq_triggered) {
      timer_irq_triggered = false;
      __enable_irq();
      main_cycle();
    } else if(ff_irq_triggered) {
      ff_irq_triggered = false;
```

```
        __enable_irq();
        handle_ff_irq();
      } else {
                __enable_irq();
        __WFE(); /* assuming that SEVONPEND in the System Control Register is
NOT set */
      }
   }
#elif D_SLEEP_VERSION == 4 // version with WFE and without IRQ synchronization
   while (true) {
      if(timer_irq_triggered) {
        timer_irq_triggered = false;
        ff_irq_triggered = false;
        handle_ff_irq();
      } else {
        __WFE(); /* assuming that SEVONPEND in the System Control Register is
NOT set */
        __WFE(); /* it is recommended that SEVONPEND in the
            System Control Register is NOT set */
      }
   }
#else  // D_SLEEP_VERSION != [1|2|3|4]
#error "Unsupported D_SLEEP_VERSION selected!"
#endif // D_SLEEP_VERSION != [1|2|3|4]
        }
```

ARM Technical Reference details:

For reference, you can consult the following:

- What is the purpose of WFI and WFE instructions and the event signals?
- Why does the processor enter standby when using WFE instruction but not when using WFI instruction?
- WIC - Wakeup using WIC
  Upon receipt of Wake-up Interrupt Controller Signal, processor immediately resumes execution of instruction. This feature is optional and when implemented usually applies to **Deep Sleep wakeup** only

The short detailed documents on the WFE and WFI instructions are:

```
    3.7.11. WFE
    Wait For Event.
    Syntax
    WFE
    Operation
    If the event register is 0, WFE suspends execution until one of the following
events occurs:
    (i)  an exception, unless masked by the exception mask registers or the current
      priority level
    (ii) an exception enters the Pending state, if SEVONPEND in the System Control
          Register is set
    (iii)  a Debug Entry request, if debug is enabled
    (iv) an event signaled by a peripheral or another processor in a multiprocessor
          system using the SEV instruction.

    If the event register is 1, WFE clears it to 0 and completes immediately.
    For more information see Power management.
    Note
    WFE is intended for power saving only. When writing software assume that WFE
might behave as NOP.
    Restrictions
    There are no restrictions.
    Condition flags
    This instruction does not change the flags.
    Examples
        WFE  ; Wait for event
```

The WFI:

```
    3.7.12. WFI
    Wait for Interrupt.
    Syntax
    WFI
    Operation
    WFI suspends execution until one of the following events occurs:
    (i)  an exception
    (ii) an interrupt becomes pending, which would preempt if PRIMASK was clear
    (iii)  a Debug Entry request, regardless of whether debug is enabled.
    Note
    WFI is intended for power saving only. When writing software assume that WFI
might behave as a NOP operation.
    Restrictions
    There are no restrictions.
    Condition flags
    This instruction does not change the flags.
    Examples
        WFI ; Wait for interrupt
```