Interrupt Programming

Last updated 6/3/21

Interrupt Programming

- Blocks that support interrupts
  - Nucleo-L476RG - Almost all blocks support interrupts
    - ADC, Timers, Inputs, SPI, …
  - Mbed only supports interrupts on a few blocks
    - InterruptIn – interrupt on pin changes
    - Timers – causes an interrupt after a specific time
    - Ticker – Causes a repeated interrupt at a defined interval
  - Mbed uses the default priorities
    - No methods in Mbed to change priorities

Interrupt Programming

- WARNING, WARNING, WARNING
  - Interrupts should be made as fast as possible
    - Other critical interrupts may be ignored during the current interrupt
    - No printf
    - No While(1)
    - No wait, thread_sleep
    - No complex processing
  - Interrupts can only see global variables
    - This is our exception to no global variables allowed
  - Interrupt service routines (ISRs)
    - Must return “void”
    - Must not have any parameters

      void my_isr(void)

Interrupt Programming

- Interrupt programming steps
  1. Define the interrupt object
  2. Attach an ISR to the object
  3. Write the ISR – remember all the warnings

Interrupt Programming

- InterruptIn Class – Input Pin Interrupts

Public Member Functions	
	InterruptIn (PinName pin)
	Create an InterruptIn connected to the specified pin. More...
	InterruptIn (PinName pin, PinMode mode)
	Create an InterruptIn connected to the specified pin, and the pin configured to the specified mode. More...
int	read ()
	Read the input, represented as 0 or 1 (int) More...
	operator int ()
	An operator shorthand for read() More...
void	rise (Callback< void()> func)
	Attach a function to call when a rising edge occurs on the input. More...
void	fall (Callback< void()> func)
	Attach a function to call when a falling edge occurs on the input. More...
void	mode (PinMode pull)
	Set the input pin mode. More...
void	enable_irq ()
	Enable IRQ. More...
void	disable_irq ()
	Disable IRQ. More...

Interrupt Programming

- Constructors

InterruptIn (PinName pin)
Create an InterruptIn connected to the specified pin. More...
InterruptIn (PinName pin, PinMode mode)
Create an InterruptIn connected to the specified pin, and the pin configured to the specified mode. More...

```
// create an interrupt object for Pin D4

InterruptIn Increment(D4);
```

Interrupt Programming

- Member Functions (Methods)

int	read ()
	Read the input, represented as 0 or 1 (int) More...
void	rise (Callback< void()> func)
	Attach a function to call when a rising edge occurs on the input. More...
void	fall (Callback< void()> func)
	Attach a function to call when a falling edge occurs on the input. More...
void	mode (PinMode pull)
	Set the input pin mode. More...
void	enable_irq ()
	Enable IRQ. More...
void	disable_irq ()
	Disable IRQ. More...

```
// and attach the isr

Increment.rise(&increment_isr);
```

Interrupt Programming

- Operator Overloads

	operator int ()
	An operator shorthand for read() More...

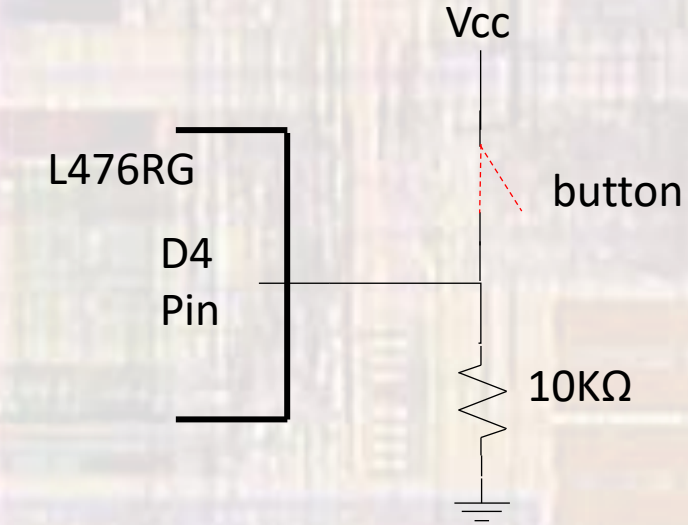
Interrupt Programming

- Simple example 1
  - Use pin D4 interrupt to increment a counter

```
////////////////////////////////////
//
// int_class_ex_1 project
//
// created 5/12/21 by tj
// rev 0
//
////////////////////////////////////
//
// interrupt example file for class
//
// interrupt based counter
//
////////////////////////////////////

#include "mbed.h"
#include <stdio.h>

#define T_WAIT 3000000  // in us - 3sec

// function prototypes (declaration)
void increment_isr(void);

// Global HARDWARE Objects
// create an interrupt object for Pin D4
InterruptIn Increment(D4);

// global variables for ISR
// note it is defined as volatile since it can
// change without main knowing it - volatile
// forces it to be read from memory each time
// instead of from a CPU register
volatile int cnt = 0;

int main(void){
    setbuf(stdout, NULL);   // disable buffering

    // splash
    printf("\n\nint_class_ex_1 - example for EE2905\n");
    printf("Using Mbed OS version %d.%d.%d\n\n",
           MBED_MAJOR_VERSION, MBED_MINOR_VERSION, MBED_PATCH_VERSION);

   // Attach the isr to the global HW object
   Increment.rise(&increment_isr);
```

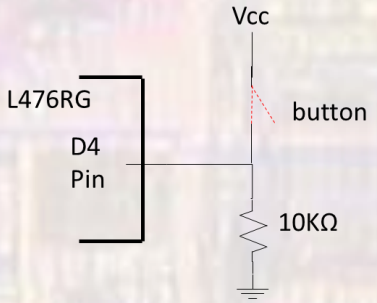
```
   // create a waiting loop
   while(1){
       wait_us(T_WAIT);
       printf("The current count is: %i\n", cnt);
   } // end while

   return 0;
}// end main
```

```
void increment_isr(void){
    // interrupt service routine for counter clk
    // increments the cnt

    cnt++;

    return;
} // end increment_isr
```



Interrupt Programming

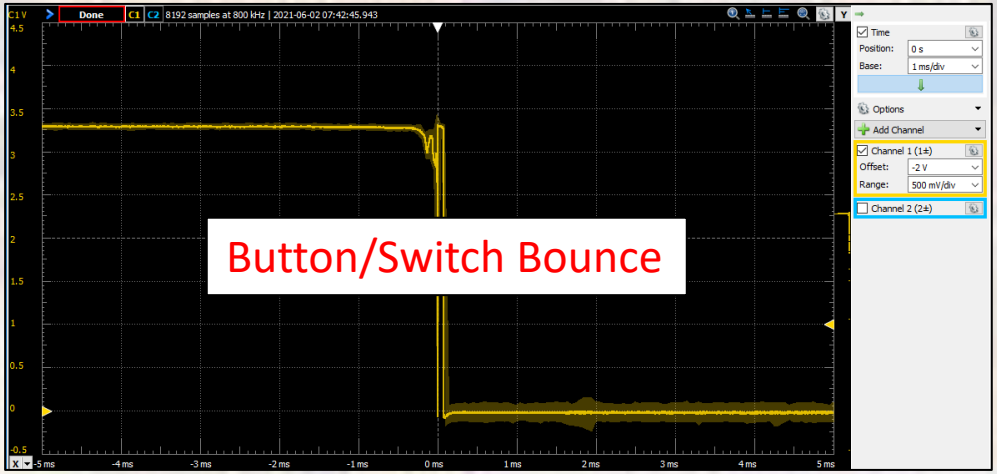
- Simple example 1
  - Use pin D4 interrupt to increment a counter




```
int_class_ex_1 - example for EE2905
Using Mbed OS version 6.11.0

The current count is: 0
The current count is: 0
The current count is: 0
The current count is: 3
The current count is: 4
The current count is: 7
The current count is: 9
The current count is: 10
The current count is: 11
The current count is: 14
The current count is: 15
The current count is: 20
The current count is: 22
The current count is: 23
```

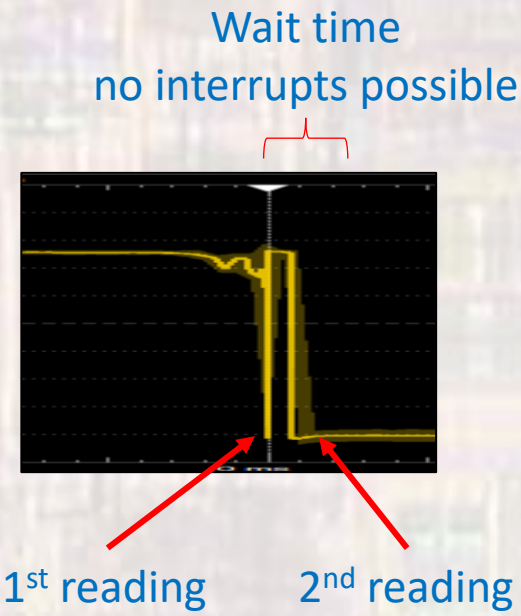
1 press each loop starting here

Some loops have more than 1 increment – WHY?



Interrupt Programming

- Simple example 2
  - Use pin D4 interrupt to increment a counter w/debounce
  - In the ISR
    - Read the pin value
    - Wait long enough for any bouncing to stop
      - Prevents additional interrupts from occuring
    - Read the pin value again
    - If the same – assume the input is valid
      - Take whatever action the ISR is supposed to take
  - Note: this is far from a perfect solution
    - Use hardware debouncing for critical situations



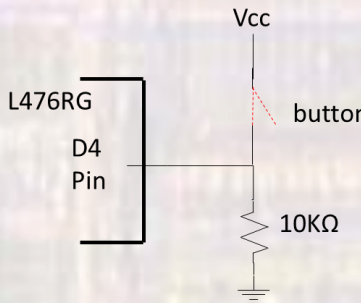
Interrupt Programming

- Simple example 2
  - Use pin D4 interrupt to [illegible]debounce

```
//////////////////////////////////////
//
// int_class_ex_2 project
//
// created 5/12/21 by tj
// rev 0
//
//////////////////////////////////////
//
// interrupt example file for class
//
// interrupt based counter with debounce
// Note - there are better ways to do this, e.g. timers
//
//////////////////////////////////////

#include "mbed.h"
#include <stdio.h>

#define T_WAIT 3000000  // in us - 3sec
#define T_BOUNCE 5000   // in us - 5ms

// function prototypes (declaration)
void increment_isr(void);

// Global HARDWARE Objects
// create an interrupt object for Pin D4
InterruptIn Increment(D4);

// global variables for ISR
// note it is defined as volatile since it can
// change without main knowing it - volatile
// forces it to be read from memory each time
// instead of from a CPU register
volatile int cnt;

int main(void){
    setbuf(stdout, NULL);   // disable buffering

    // splash
    printf("\n\nint_class_ex_2 - example for EE2905\n");
    printf("Using Mbed OS version %d.%d.%d\n\n",
           MBED_MAJOR_VERSION, MBED_MINOR_VERSION, MBED_PATCH_VERSION);

   // attach the isr
   Increment.rise(&increment_isr);
```

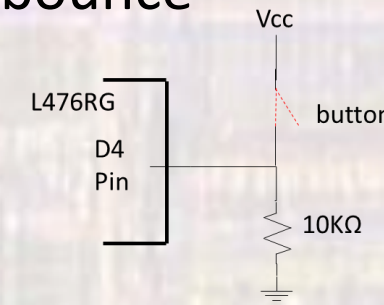
```
   // initialize the count and valid values
   cnt = 0;


   // create a waiting loop
   while(1){
       wait_us(T_WAIT);
       printf("The current count is: %i\n", cnt);
   } // end while

   return 0;
}// end main
```

```
void increment_isr(void){
    // interrupt service routine for counter clk
    // debounces the pin and increments the cnt

    //////////////////////////////////////////
    // Debounce section
    uint8_t pinval_1;
    uint8_t pinval_2;
    uint8_t valid;

    // first check
    pinval_1 = Increment.read();

    // allow bounce to complete
    wait_us(T_BOUNCE);  // debounce
                        // terrible way to solve the problem

    // second check
    pinval_2 = Increment.read();

    // set "valid" if not a bounce and value is 1 (pushed)
    if(pinval_1 == pinval_2)
        valid = pinval_1;
    //
    //////////////////////////////////////

    //////////////////////////////////////
    // count update section
    if(valid)
        cnt++;
    //
    //////////////////////////////////////


    return;
} // end increment_isr
```



Interrupt Programming

- Simple example 2
  - Use pin D4 interrupt to increment a counter w/debounce




```
int_class_ex_2 - example for EE2905
Using Mbed OS version 6.11.0

The current count is: 0
The current count is: 0
The current count is: 0
The current count is: 0
The current count is: 1
The current count is: 1
The current count is: 1
The current count is: 1
The current count is: 2
The current count is: 2
The current count is: 4
The current count is: 4
The current count is: 4
The current count is: 4
The current count is: 7
The current count is: 7
The current count is: 7
The current count is: 11
The current count is: 11
The current count is: 11
The current count is: 16
The current count is: 16
```

1 press

1 press

2 presses

3 presses

4 presses

5 presses

Interrupt Programming

- Simple example 3
  - Estimate the context latency for an interrupt
    - Create a main loop that forces a digital output low
    - Use the ISR to toggle the output pin high
    - Use a digital square wave input signal
    - Compare the input signal to the output waveform
    - The time from input rise to output rise is an estimate of the ISR entry latency
    - The width of the output high pulse is an estimate of the ISR exit latency

Interrupt Programming

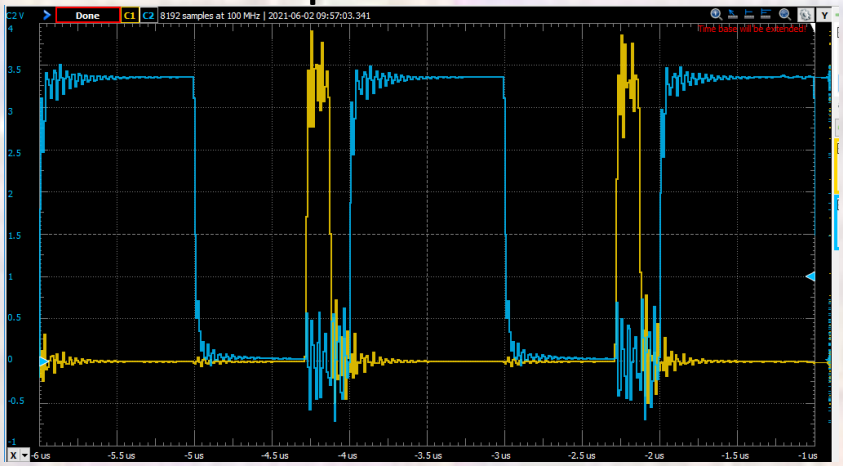
- Simple example 3
  - Estimate the Interrupt context switch times

```
////////////////////////////////////
//
// int_class_ex_3 project
//
// created 5/12/21 by tj
// rev 0
//
////////////////////////////////////
//
// interrupt example file for class
//
// interrupt latency estimate
//
////////////////////////////////////

#include "mbed.h"
#include <stdio.h>

// function prototypes (declaration)
void int_in_isr(void);

// Global HARDWARE Objects
// create an interrupt object for Pin D4
InterruptIn Int_in(D4);
// create an input object for pin D6
DigitalOut Int_out(D6);

int main(void){
    setbuf(stdout, NULL);   // disable buffering

    // splash
    printf("\n\nint_class_ex_3 - example for EE2905\n");
    printf("Using Mbed OS version %d.%d.%d\n\n",
           MBED_MAJOR_VERSION, MBED_MINOR_VERSION, MBED_PATCH_VERSION);

   // Attach the isr to the global HW object
   Int_in.rise(&int_in_isr);

   // Initialize the output
   Int_out.write(0);

   // create a waiting loop and force Int_out to 0
   while(1){
       Int_out = 0;
   } // end while

   return 0;
}// end main
```

```
void int_in_isr(void){
    // interrupt service routine for int_in
    // toggle the output

    Int_out = !Int_out;

    return;
} // end int_in_isr
```

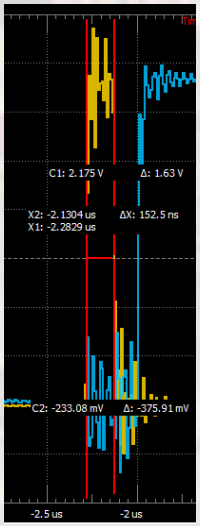
Interrupt Programming

- Simple example 3
  - Estimate the Interrupt context switch times



Special considerations with the test ???

Event to ISR Action
1.7 us (136 clks)



ISR Action to Main action
152 ns (12clks)

Why the big difference ???