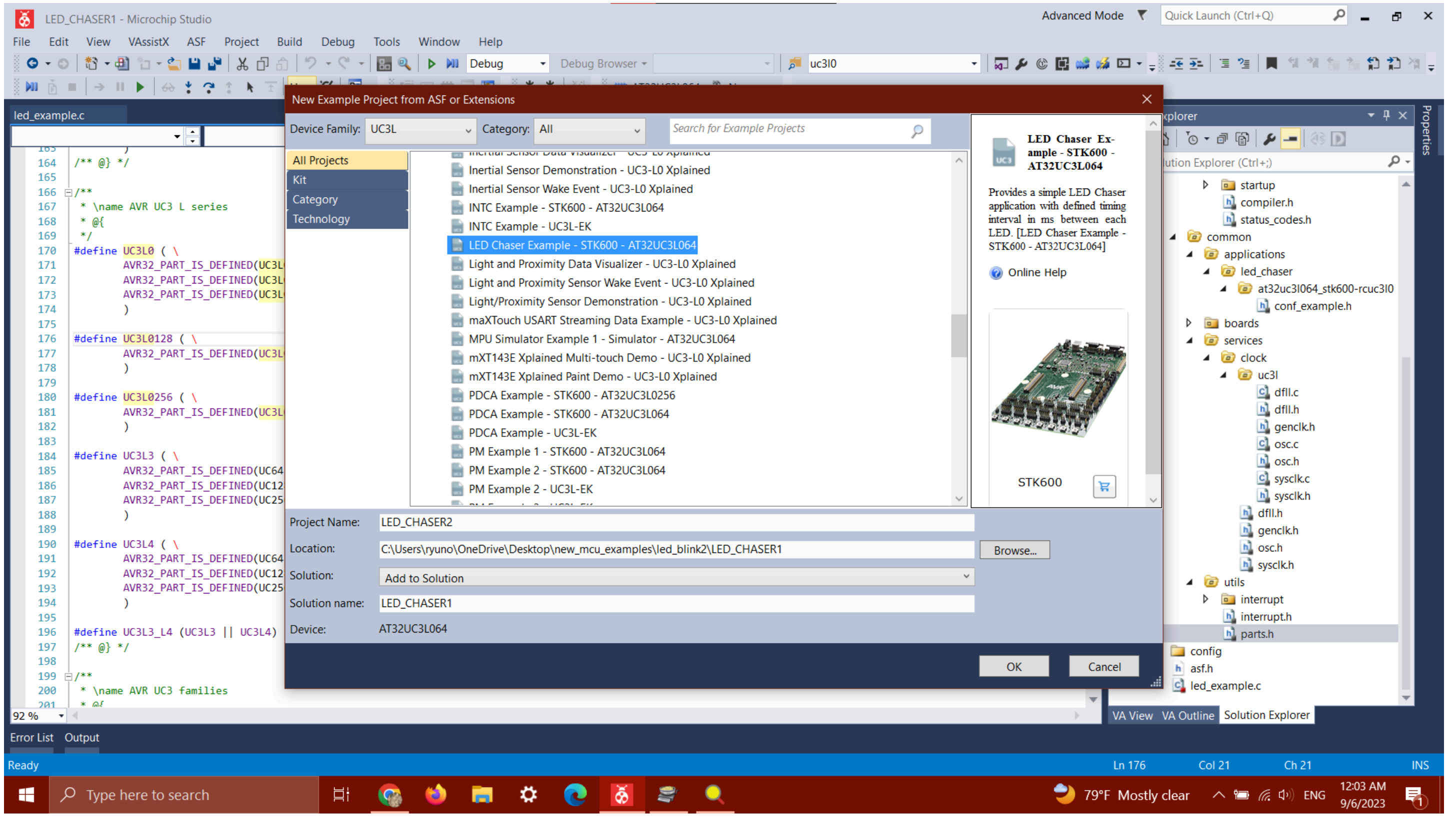
9/5/23:

- Started mouse scratch notes to think continuously just like at work
- Looking at led_example project made for the AT32UCL064 on the STK600 devboard



New Example Project from ASF or Extensions

Device Family: UC3L | Category: All

- Inertial Sensor Demonstration - UC3-L0 Xplained
- Inertial Sensor Wake Event - UC3-L0 Xplained
- INTC Example - STK600 - AT32UC3L064
- INTC Example - UC3L-EK
- LED Chaser Example - STK600 - AT32UC3L064
- Light and Proximity Data Visualizer - UC3-L0 Xplained
- Light and Proximity Sensor Wake Event - UC3-L0 Xplained
- Light/Proximity Sensor Demonstration - UC3-L0 Xplained
- maXTouch USART Streaming Data Example - UC3-L0 Xplained
- MPU Simulator Example 1 - Simulator - AT32UC3L064
- mXT143E Xplained Multi-touch Demo - UC3-L0 Xplained
- mXT143E Xplained Paint Demo - UC3-L0 Xplained
- PDCA Example - STK600 - AT32UC3L0256
- PDCA Example - STK600 - AT32UC3L064
- PDCA Example - UC3L-EK
- PM Example 1 - STK600 - AT32UC3L064
- PM Example 2 - STK600 - AT32UC3L064
- PM Example 2 - UC3L-EK

LED Chaser Example - STK600 - AT32UC3L064

Provides a simple LED Chaser application with defined timing interval in ms between each LED. [LED Chaser Example - STK600 - AT32UC3L064]

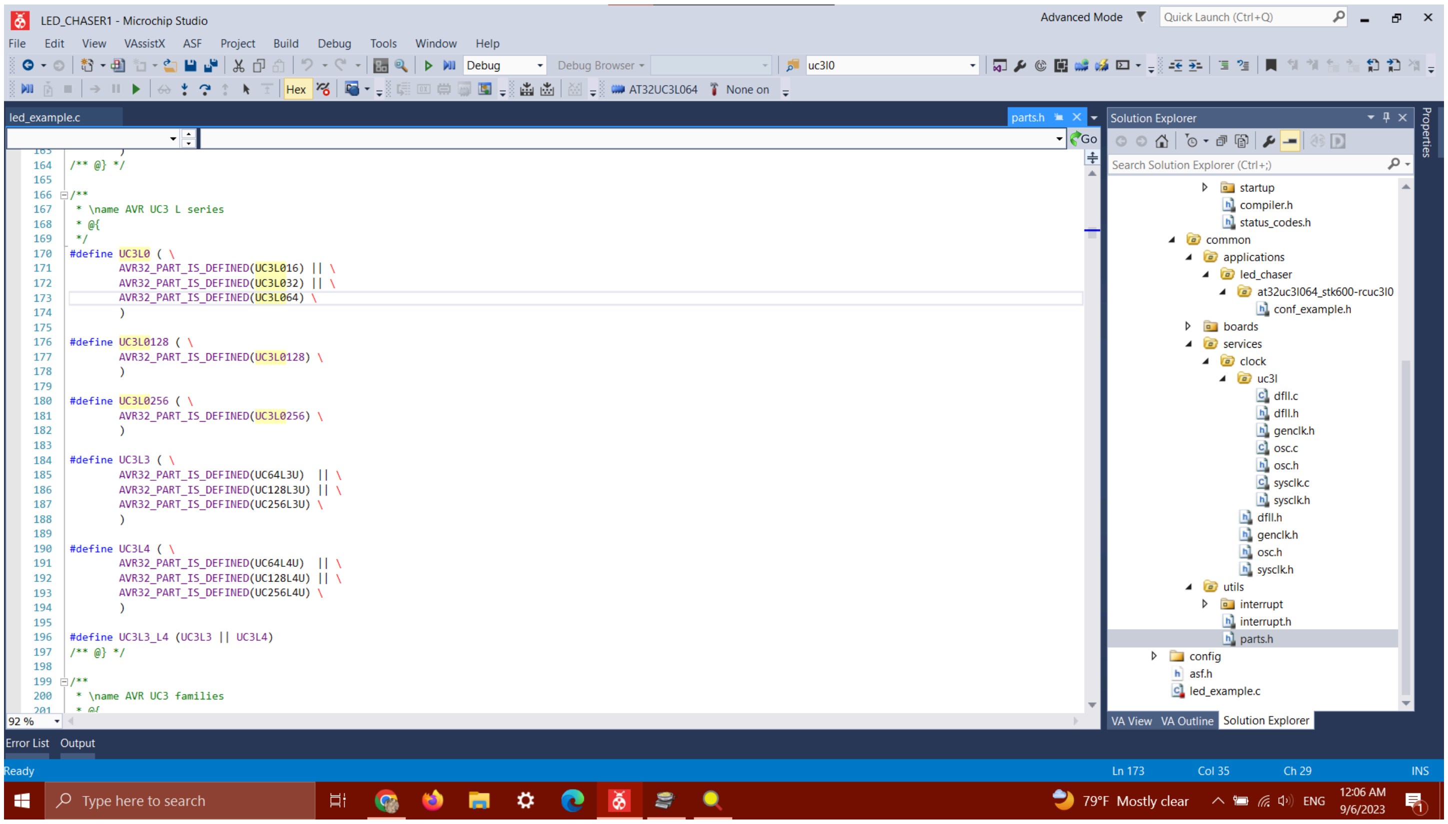
Online Help

STK600

Project Name: LED_CHASER2
Location: C:\Users\ryuno\OneDrive\Desktop\new_mcu_examples\led_blink2\LED_CHASER1
Solution: Add to Solution
Solution name: LED_CHASER1
Device: AT32UC3L064

OK | Cancel

- It seems like UC3L0164 and UC3L0256 are the only two mcus that are defined, but not sure



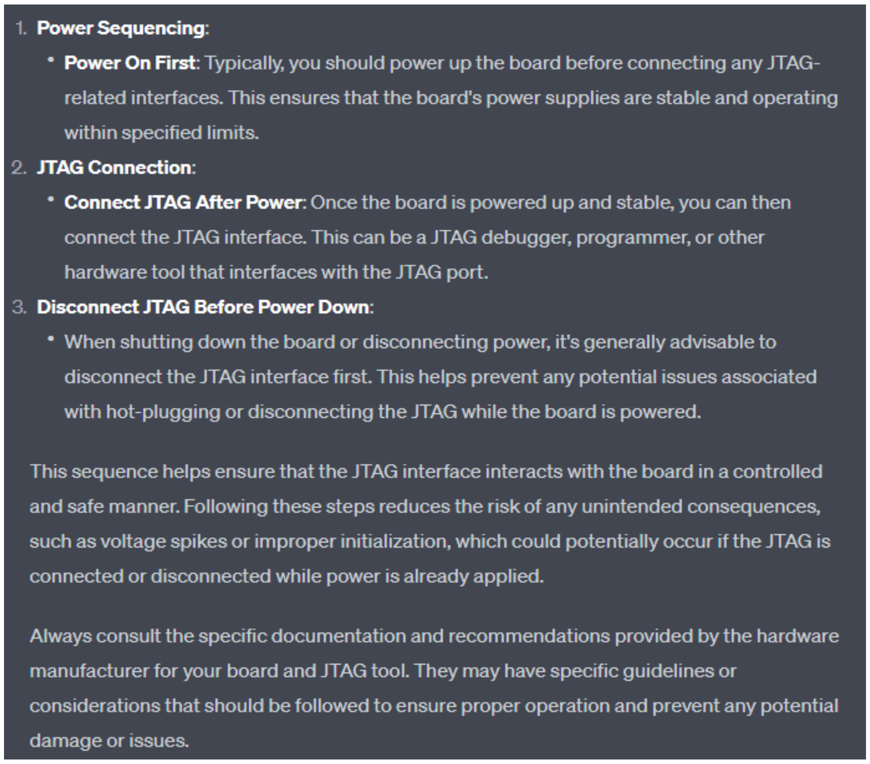
led_example.c

```
/** @} */

/**
 * \name AVR UC3 L series
 * @{
 */
#define UC3L0 ( \
        AVR32_PART_IS_DEFINED(UC3L016) || \
        AVR32_PART_IS_DEFINED(UC3L032) || \
        AVR32_PART_IS_DEFINED(UC3L064) \
        )

#define UC3L0128 ( \
        AVR32_PART_IS_DEFINED(UC3L0128) \
        )

#define UC3L0256 ( \
        AVR32_PART_IS_DEFINED(UC3L0256) \
        )

#define UC3L3 ( \
        AVR32_PART_IS_DEFINED(UC64L3U)  || \
        AVR32_PART_IS_DEFINED(UC128L3U) || \
        AVR32_PART_IS_DEFINED(UC256L3U) \
        )

#define UC3L4 ( \
        AVR32_PART_IS_DEFINED(UC64L4U)  || \
        AVR32_PART_IS_DEFINED(UC128L4U) || \
        AVR32_PART_IS_DEFINED(UC256L4U) \
        )

#define UC3L3_L4 (UC3L3 || UC3L4)
/** @} */

/**
 * \name AVR UC3 families
 * @{
```

Next steps:

- Try to upload code to new mcu powered w/ lipo battery

9/11/23:

- Encountering issues uploading to mcu consistently
  - PCB is missing reset connection
  - Motors aren't fully center on PCB vertically
- Try debugging on protoboard:
- Ideas for home project
  - ■ Monitor power signals on repeated programming attempts to confirm that power is stable (mcu on protoboard version)
    - ● Maybe a quick for switching regulators is that it can swing up and down outside of just the switching it does aggressively?
    - ● Order in which you power things might be important- maybe it'd work consistently if you power the circuit on its own first, and then plug in usb
      - ○ We've been plugging in power while usb and jtag are already plugged in every single time, so maybe try plugging in jtag after
      - ○ 

1. **Power Sequencing**:
   - **Power On First**: Typically, you should power up the board before connecting any JTAG-related interfaces. This ensures that the board's power supplies are stable and operating within specified limits.
2. **JTAG Connection**:
   - **Connect JTAG After Power**: Once the board is powered up and stable, you can then connect the JTAG interface. This can be a JTAG debugger, programmer, or other hardware tool that interfaces with the JTAG port.
3. **Disconnect JTAG Before Power Down**:
   - When shutting down the board or disconnecting power, it's generally advisable to disconnect the JTAG interface first. This helps prevent any potential issues associated with hot-plugging or disconnecting the JTAG while the board is powered.

This sequence helps ensure that the JTAG interface interacts with the board in a controlled and safe manner. Following these steps reduces the risk of any unintended consequences, such as voltage spikes or improper initialization, which could potentially occur if the JTAG is connected or disconnected while power is already applied.

Always consult the specific documentation and recommendations provided by the hardware manufacturer for your board and JTAG tool. They may have specific guidelines or considerations that should be followed to ensure proper operation and prevent any potential damage or issues.

      - ○
      - ○ Let's follow this going forward this week
  - ■ Look into checking for debugwire debugging vs jtag configuration settings
  - ■ Add capacitors to setup matching pcb schematic
    - ● Apply "capacitor staggering" and add a bunch of different ceramic caps to see if it solves the issue
    - ● Smaller val caps may be better provided JTAG signals are fast
  - ■ Switch usb cable maybe
  - ■ Use oscope to monitor each of the jtag signals one by one to figure out which one is causing issues

        - Eventually get logic analyzer
    - Maybe just email microchip or something if this is a well known issue w/ the mcu
        - Look into microchip sources of official support
    - Check datasheet and schematic checklist doc again for any missing circuitry

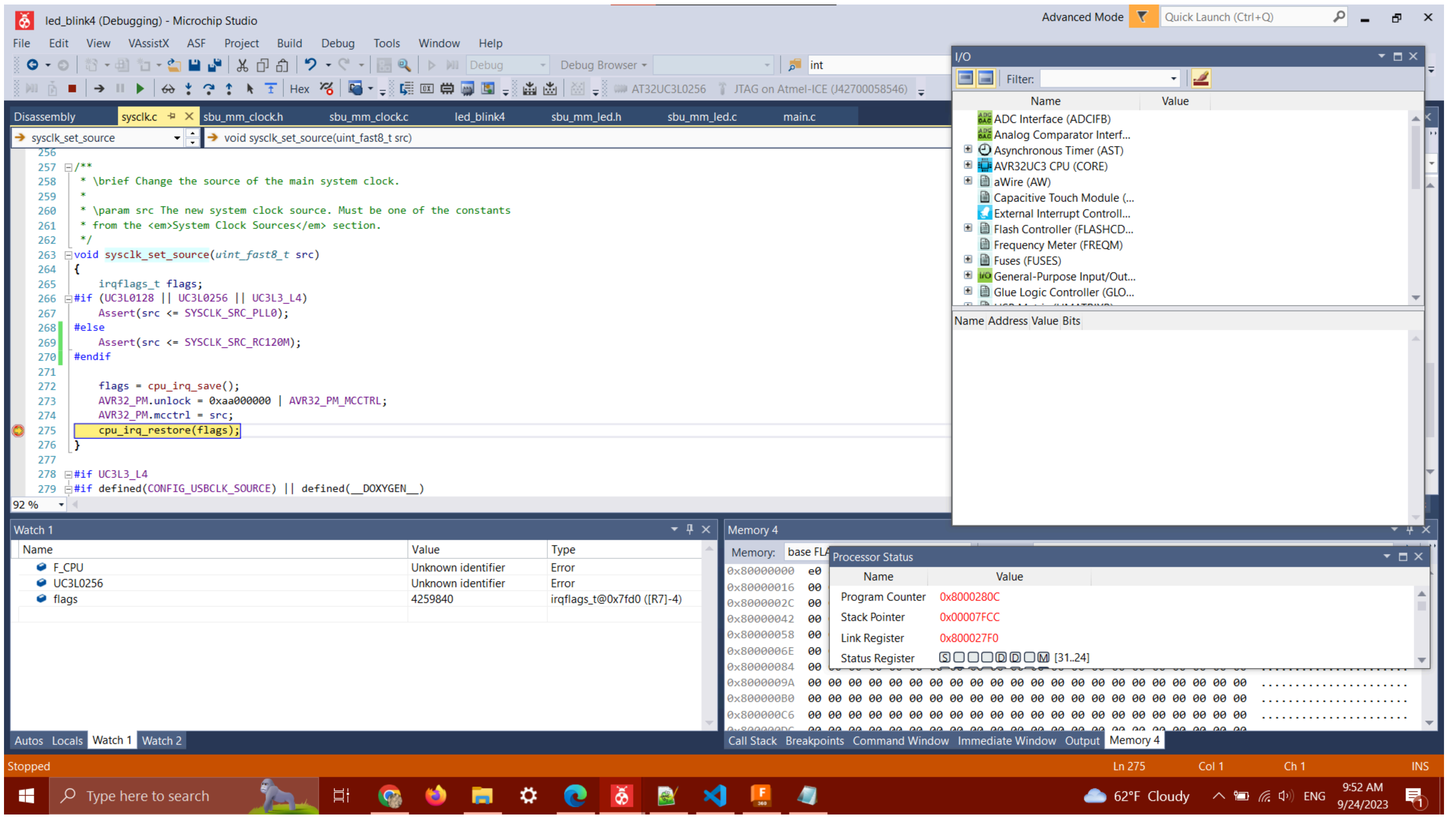
- Added a bunch of ceramic caps
- making sure to power board first and then connect JTAG
- replaced usb micro b cable to working ieee cable
- Added vddanana and gndana cables
    - Erasing chip works
    - Programming chip doesn’t work
    - Writing fuses doesn’t work, but all fuse values should be ok

- Added 10uF electrolytic caps for decoupling
    - Flashing to chip doesn’t work
    - Writing to fuses doesn’t work
    - …but single step debugging works for some reason

- …writing to flash success
- Single stepping works consistently now
    - Modified code, power cycled, removed usb, single stepping still seems to work
    - Looks like it needed more decoupling capacitance

    - Actual pcb has 10uF ceramic caps on there so it should be fine, but if it fails we’ll edit to add electrolytic or polymer caps for decoupling 💀

9/24/23:



led_blink4 (Debugging) - Microchip Studio

```
/**
 * \brief Change the source of the main system clock.
 *
 * \param src The new system clock source. Must be one of the constants
 * from the <em>System Clock Sources</em> section.
 */
void sysclk_set_source(uint_fast8_t src)
{
	irqflags_t flags;
#if (UC3L0128 || UC3L0256 || UC3L3_L4)
	Assert(src <= SYSCLK_SRC_PLL0);
#else
	Assert(src <= SYSCLK_SRC_RC120M);
#endif

	flags = cpu_irq_save();
	AVR32_PM.unlock = 0xaa000000 | AVR32_PM_MCCTRL;
	AVR32_PM.mcctrl = src;
	cpu_irq_restore(flags);
}

#if UC3L3_L4
#if defined(CONFIG_USBCLK_SOURCE) || defined(__DOXYGEN__)
```

Watch 1

Name	Value	Type
F_CPU	Unknown identifier	Error
UC3L0256	Unknown identifier	Error
flags	4259840	irqflags_t@0x7fd0 ([R7]-4)

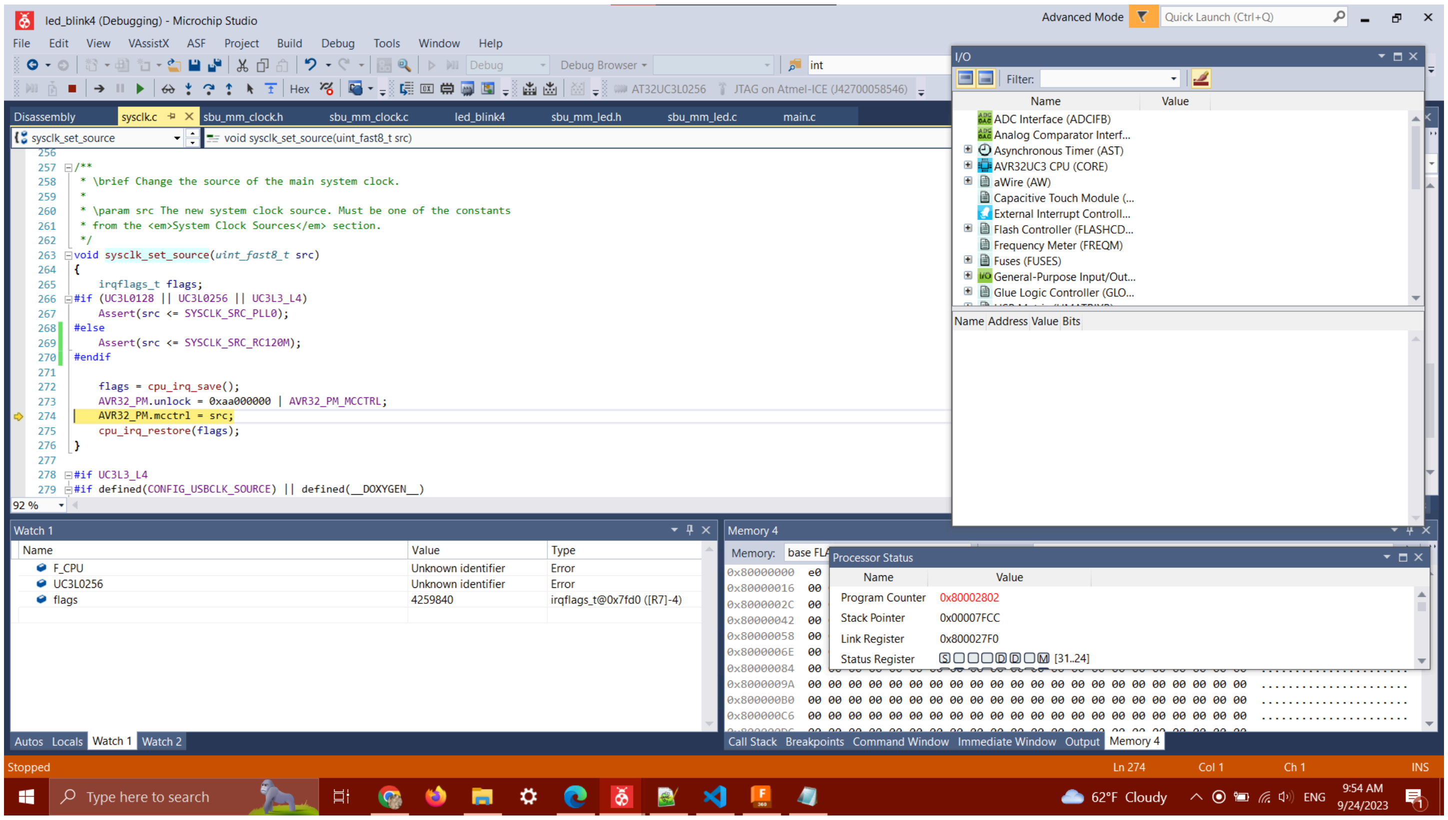
Processor Status

Name	Value
Program Counter	0x8000280C
Stack Pointer	0x00007FCC
Link Register	0x800027F0
Status Register	[31..24]

Memory 4

```
0x80000000  e0
0x80000016  00
0x8000002C  00
0x80000042  00
0x80000058  00
0x8000006E  00
0x80000084  00
0x8000009A  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ......................
0x800000B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ......................
0x800000C6  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ......................
```

Stopped Ln 275 Col 1 Ch 1 INS



led_blink4 (Debugging) - Microchip Studio

```
/**
 * \brief Change the source of the main system clock.
 *
 * \param src The new system clock source. Must be one of the constants
 * from the <em>System Clock Sources</em> section.
 */
void sysclk_set_source(uint_fast8_t src)
{
	irqflags_t flags;
#if (UC3L0128 || UC3L0256 || UC3L3_L4)
	Assert(src <= SYSCLK_SRC_PLL0);
#else
	Assert(src <= SYSCLK_SRC_RC120M);
#endif

	flags = cpu_irq_save();
	AVR32_PM.unlock = 0xaa000000 | AVR32_PM_MCCTRL;
	AVR32_PM.mcctrl = src;
	cpu_irq_restore(flags);
}

#if UC3L3_L4
#if defined(CONFIG_USBCLK_SOURCE) || defined(__DOXYGEN__)
```

Watch 1

Name	Value	Type
F_CPU	Unknown identifier	Error
UC3L0256	Unknown identifier	Error
flags	4259840	irqflags_t@0x7fd0 ([R7]-4)

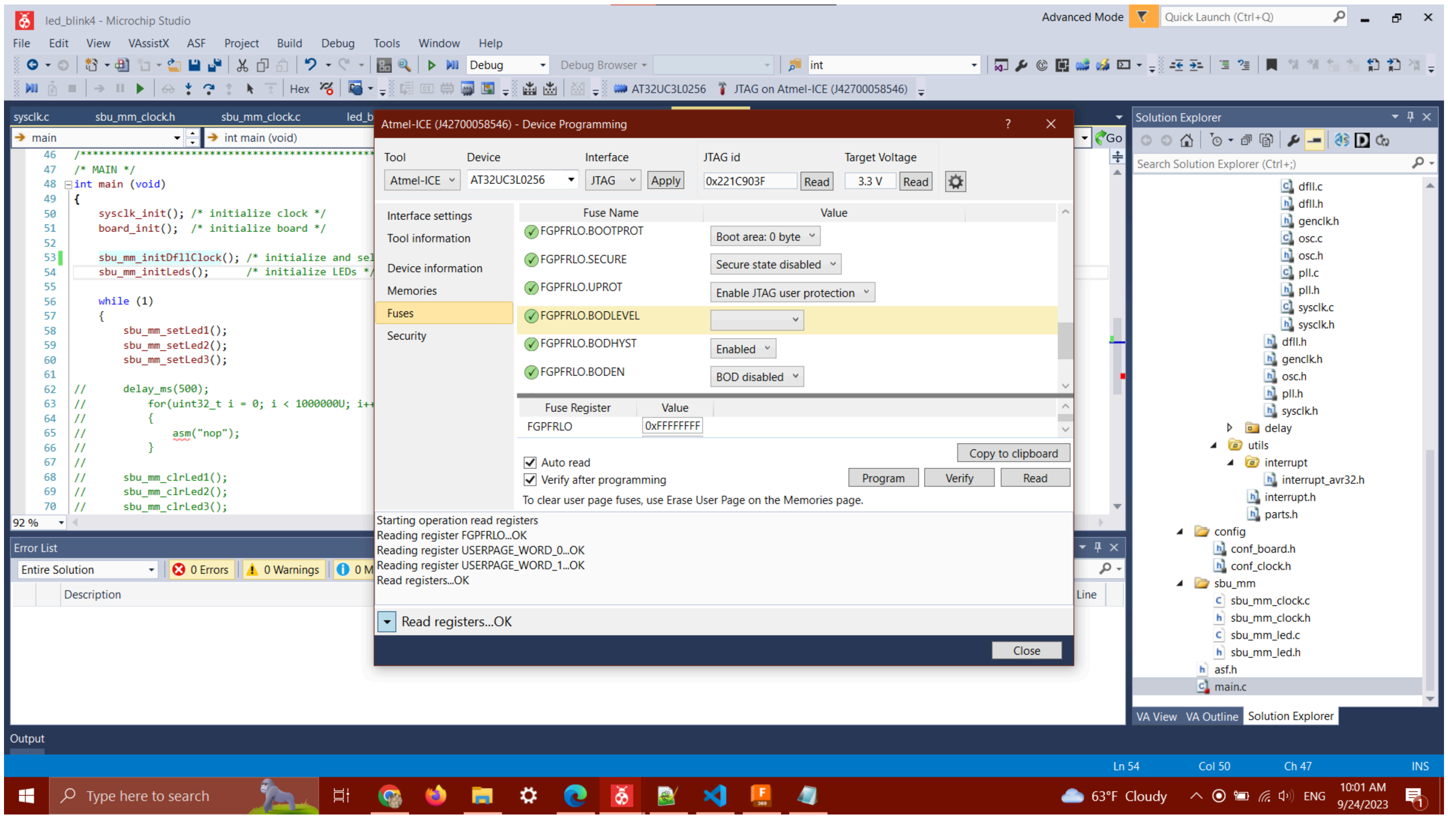
Processor Status

Name	Value
Program Counter	0x80002802
Stack Pointer	0x00007FCC
Link Register	0x800027F0
Status Register	[31..24]

Memory 4

```
0x80000000  e0
0x80000016  00
0x8000002C  00
0x80000042  00
0x80000058  00
0x8000006E  00
0x80000084  00
0x8000009A  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ......................
0x800000B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ......................
0x800000C6  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ......................
```

Stopped Ln 274 Col 1 Ch 1 INS



```
/*******************************************
/* MAIN */
int main (void)
{
    sysclk_init(); /* initialize clock */
    board_init();  /* initialize board */

    sbu_mm_initDfllClock(); /* initialize and sel
    sbu_mm_initLeds();      /* initialize LEDs */

    while (1)
    {
        sbu_mm_setLed1();
        sbu_mm_setLed2();
        sbu_mm_setLed3();

//      delay_ms(500);
//          for(uint32_t i = 0; i < 1000000U; i++
//          {
//              asm("nop");
//          }
//
//      sbu_mm_clrLed1();
//      sbu_mm_clrLed2();
//      sbu_mm_clrLed3();
```

Atmel-ICE (J42700058546) - Device Programming

Tool	Device	Interface	JTAG id	Target Voltage
Atmel-ICE	AT32UC3L0256	JTAG	0x221C903F	3.3 V

Fuse Name	Value
FGPFRLO.BOOTPROT	Boot area: 0 byte
FGPFRLO.SECURE	Secure state disabled
FGPFRLO.UPROT	Enable JTAG user protection
FGPFRLO.BODLEVEL	
FGPFRLO.BODHYST	Enabled
FGPFRLO.BODEN	BOD disabled

Fuse Register	Value
FGPFRLO	0xFFFFFFFF

☑ Auto read
☑ Verify after programming

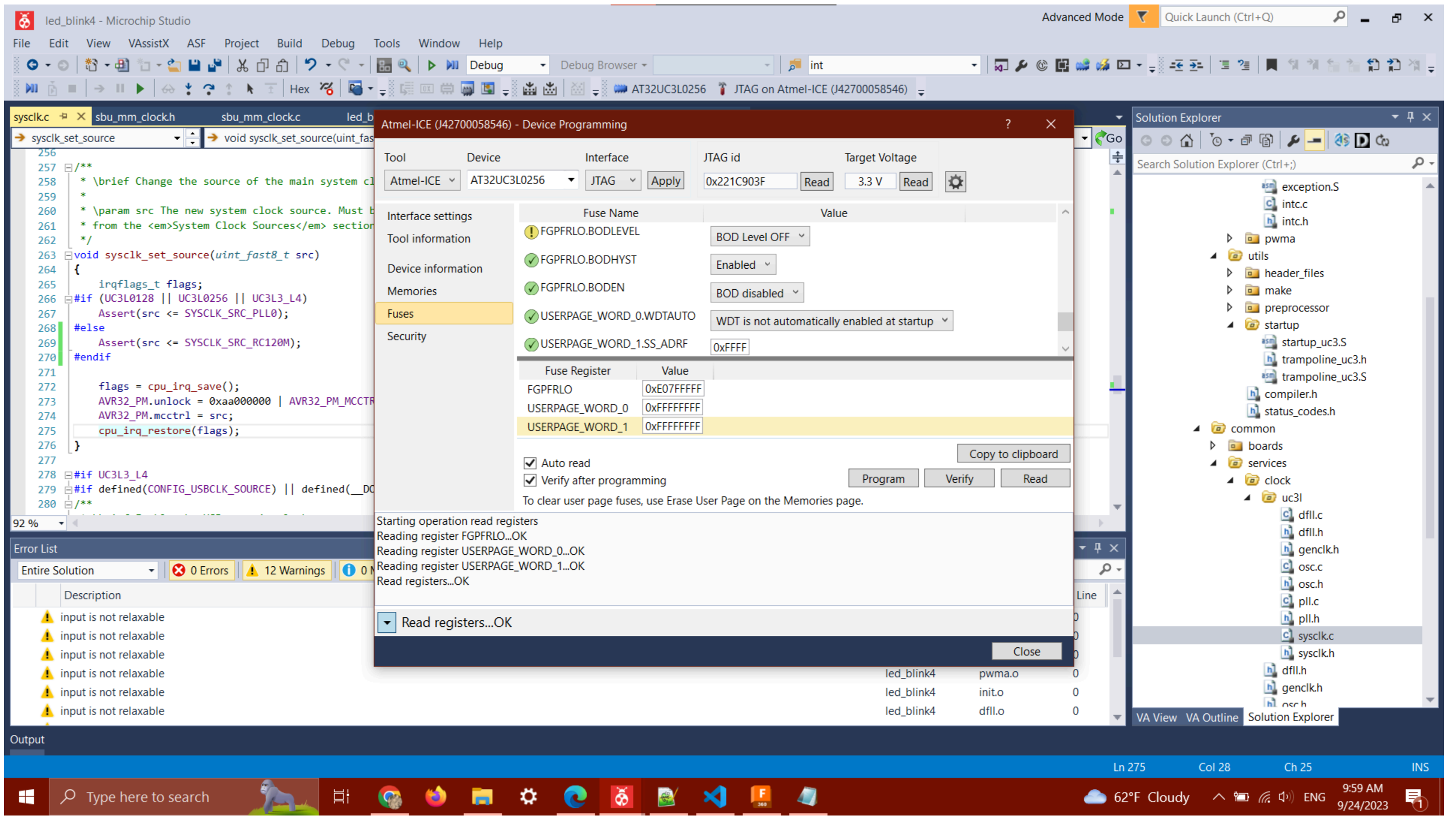
To clear user page fuses, use Erase User Page on the Memories page.

```
Starting operation read registers
Reading register FGPFRLO...OK
Reading register USERPAGE_WORD_0...OK
Reading register USERPAGE_WORD_1...OK
Read registers...OK
```

Read registers...OK

Error List: Entire Solution — 0 Errors — 0 Warnings

Ln 54 Col 50 Ch 47 INS — 10:01 AM 9/24/2023



```
/**
 * \brief Change the source of the main system cl
 *
 * \param src The new system clock source. Must b
 * from the <em>System Clock Sources</em> section
 */
void sysclk_set_source(uint_fast8_t src)
{
    irqflags_t flags;
#if (UC3L0128 || UC3L0256 || UC3L3_L4)
    Assert(src <= SYSCLK_SRC_PLL0);
#else
    Assert(src <= SYSCLK_SRC_RC120M);
#endif

    flags = cpu_irq_save();
    AVR32_PM.unlock = 0xaa000000 | AVR32_PM_MCCTR
    AVR32_PM.mcctrl = src;
    cpu_irq_restore(flags);
}

#if UC3L3_L4
#if defined(CONFIG_USBCLK_SOURCE) || defined(__DO
/**
```

Atmel-ICE (J42700058546) - Device Programming

Tool	Device	Interface	JTAG id	Target Voltage
Atmel-ICE	AT32UC3L0256	JTAG	0x221C903F	3.3 V

Fuse Name	Value
FGPFRLO.BODLEVEL	BOD Level OFF
FGPFRLO.BODHYST	Enabled
FGPFRLO.BODEN	BOD disabled
USERPAGE_WORD_0.WDTAUTO	WDT is not automatically enabled at startup
USERPAGE_WORD_1.SS_ADRF	0xFFFF

Fuse Register	Value
FGPFRLO	0xE07FFFFF
USERPAGE_WORD_0	0xFFFFFFFF
USERPAGE_WORD_1	0xFFFFFFFF

☑ Auto read
☑ Verify after programming

To clear user page fuses, use Erase User Page on the Memories page.

```
Starting operation read registers
Reading register FGPFRLO...OK
Reading register USERPAGE_WORD_0...OK
Reading register USERPAGE_WORD_1...OK
Read registers...OK
```

Read registers...OK

Error List: Entire Solution — 0 Errors — 12 Warnings

Description	Project	File	Line
input is not relaxable			0
input is not relaxable			0
input is not relaxable			0
input is not relaxable	led_blink4	pwma.o	0
input is not relaxable	led_blink4	init.o	0
input is not relaxable	led_blink4	dfll.o	0

Ln 275 Col 28 Ch 25 INS — 9:59 AM 9/24/2023

Lmao what the fuck is wrong with the new chip

9/26/23:

- ~~figure out how to reduce PWM frequency to allow more time for the rest of the application~~
- ~~confirm that set top function works w/ PWM interface~~
- translate arduino code and check if code works
    - ~~Make wheel turn one rotation~~
    - ~~make mouse move a single square~~
    - ~~make mouse make rough turns~~
    - make mouse calibrate wall detection thresholds
        - make mouse detect walls on left, right, and front using sensor values
- make mouse wall follow
- make mouse turn smoothly with PID instead of just counting encoder values and waiting for the two to match
- ~~add g_ to global variables~~
- ~~wrap init functions in mouse code into a single function to avoid tampering~~
- Try breaking loop down in rough functions for better rough turns and movement
- Try accelerating/decelerating maybe?
- Check mouse maze specs to confirm that walls are 18cm and pillars are 1.2cm