ECE 498: Design for the Internet of Things

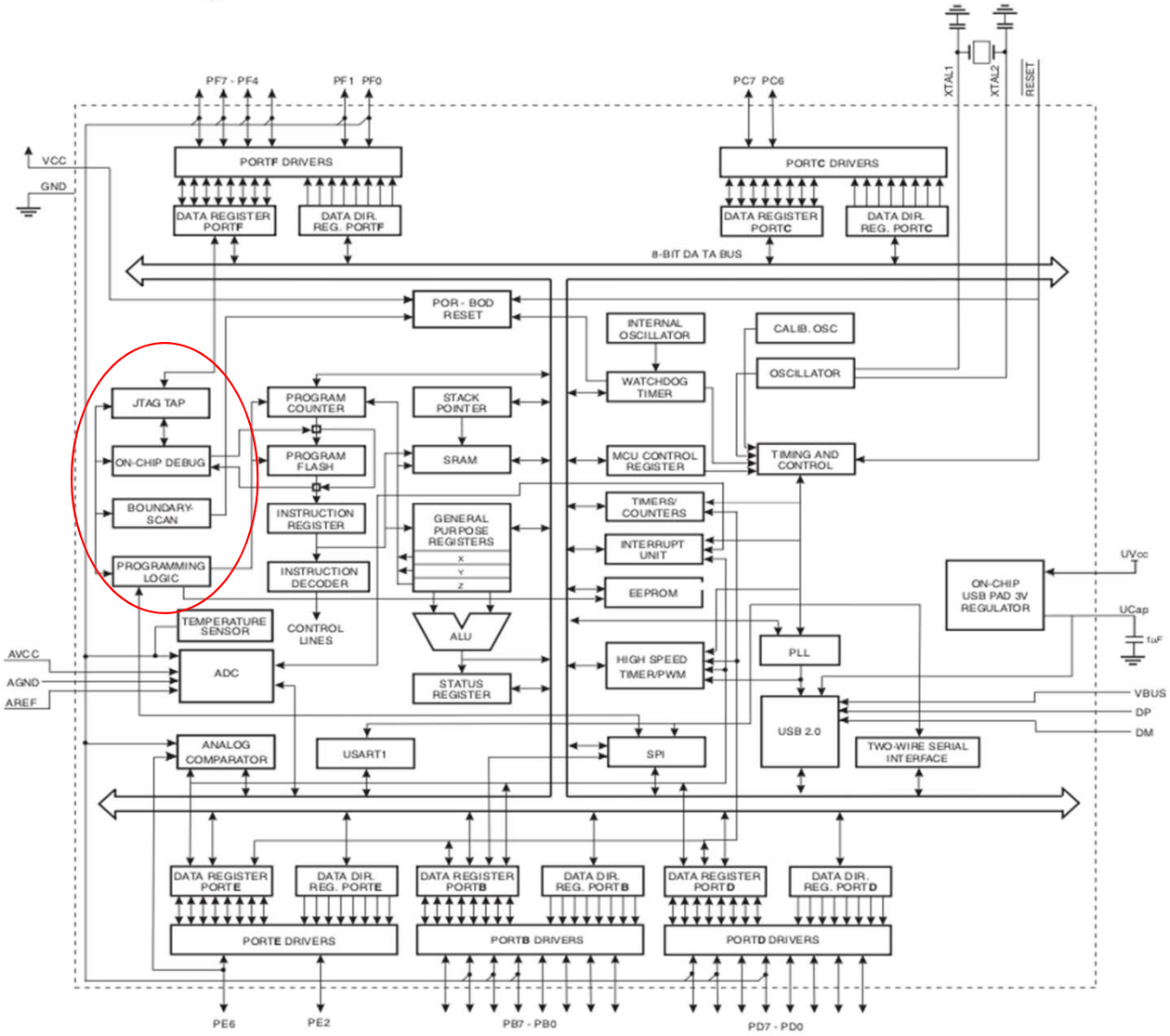
Debugging and JTAG

Dr. Jason Forsyth
Department of Engineering
James Madison University

How to Debug Your Processor?

- Print statements from the serial port are useful for basic applications
- More advanced debugging (break points, examining variable contents..etc) require an On-Chip Debugging Systems.
- Atmel 32u4 support JTAG via the SPI interface to allow direct control of the processor.

Figure 2-1. Block Diagram



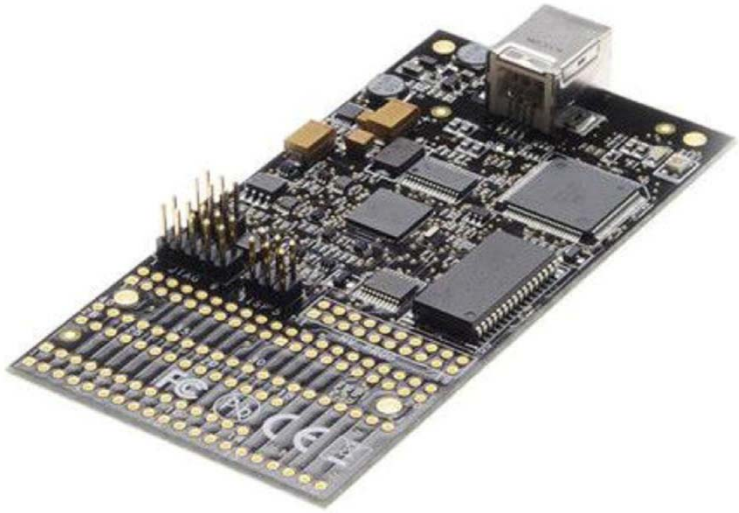
JTAG Enables Breakpoints

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(10, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);              // wait for a second
  digitalWrite(10, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);              // wait for a second
}
```

```
// Weak empty variant initialization function.
// May be redefined by variant files.
void initVariant() __attribute__((weak));
void initVariant() { }

int main(void)
{
    init();

    initVariant();

#if defined(USBCON)
    USBDevice.attach();
#endif

    setup();

    for (;;) {
        loop();
        if (serialEventRun) serialEventRun();
    }

    return 0;
}
```

Programmer / Debugger Required

AVR ISP: Supports ISP only. $25

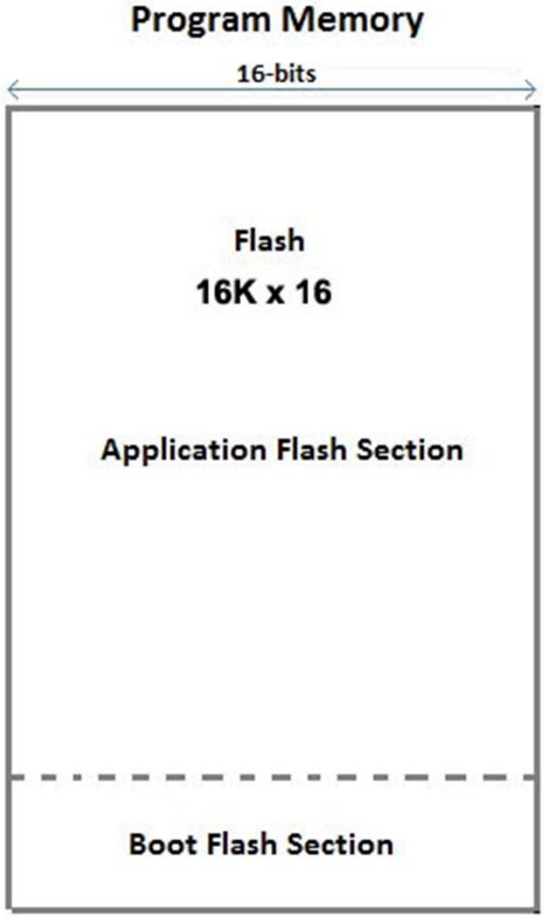


AVR Dragon: Supports ISP and JTAG only. $80

Still “programmable” with out these devices

- Purchasing programmers can be expensive when you simply want to upload code. Your $25 Arduino now costs $50…etc.

- Microprocessors support a *bootloader* to initialize device and/or load new programs. Consider as mini-program before your program launches.



Caterina Bootloader

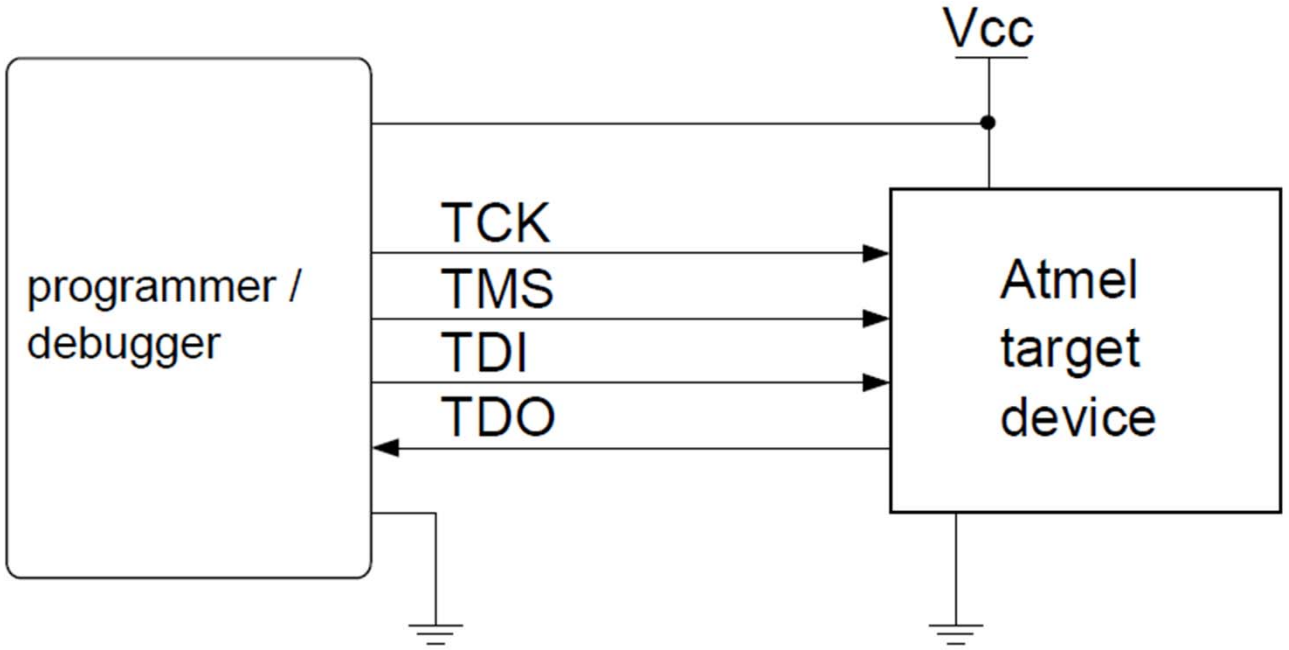
- Your Arduino's are programming via SPI because of the bootloader
- After reset is waits several seconds for new program to be loaded then launches your sketch.
- Can be disabled to avoid unwanted programming (think jailbreaking)
- Specific bootloader for your Itsy Bitsy is the Adafruit Caterina 3.3V Loader (link)

```
123  int main(void)
124  {
125      /* Setup hardware required for the bootloader */
126      SetupHardware();
127
128      /* Enable global interrupts so that the USB stack can function */
129      sei();
130
131      Timeout = 0;
132
133      while (RunBootloader)
134      {
135              CDC_Task();
              USB_USBTask();
              /* Time out and start the sketch if one is present */
              if (Timeout > TIMEOUT_PERIOD)
                      RunBootloader = false;

              LEDPulse();

      }

      /* Disconnect from the host - USB interface will be reset later along with the AVR */
      USB_Detach();

      /* Jump to beginning of application space to run the sketch - do not reset */
      StartSketch();
```

AVR Tools and Bootloader

- Cannot use the AVR tools and the bootloader at the same time
- AVR ISP and Dragon will program ALL flash memory, removing the bootloader.
- If you program with ISP or Dragon then 32u4 will no loader show up in Arduino interface
- Bootloader can be restored within Atmel Studio

Setting Up JTAG via the SPI Interface



Must establish connection between the Programmer and the “Target Device”.

First Need to Connect Via SPI to Setup JTAG

- Each “squid” pin is labeled with a number. For each squid pin on the table below connect it to the corresponding Itsy pin. If there is no associated target pin then do not connect.

Table 4-11. Atmel-ICE SPI Pin Mapping

Itsy Connections

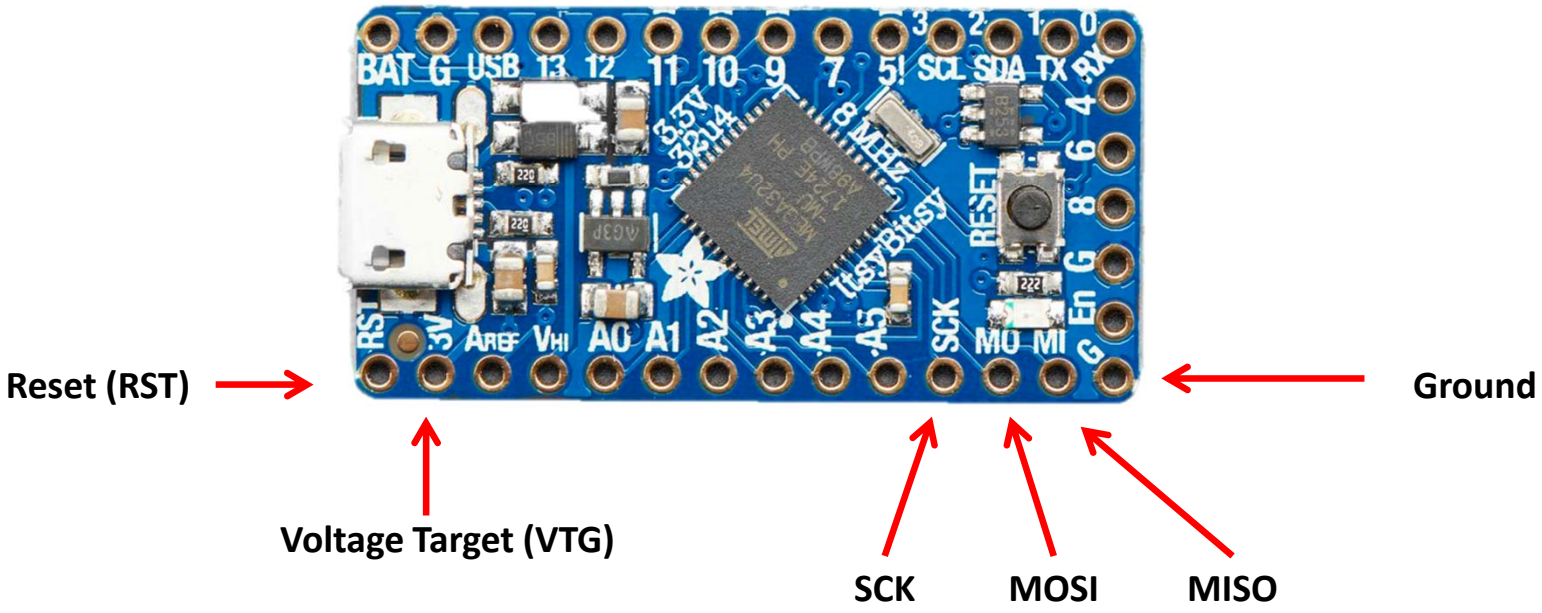
Atmel-ICE AVR port pins	Target pins	Mini-squid pin
Pin 1 (TCK)	SCK	1
Pin 2 (GND)	GND	2
Pin 3 (TDO)	MISO	3
Pin 4 (VTG)	VTG	4
Pin 5 (TMS)		5
Pin 6 (nSRST)	/RESET	6
Pin 7 (not connected)		7
Pin 8 (nTRST)		8
Pin 9 (TDI)	MOSI	9
Pin 10 (GND)		0

SPI Pin Connections

SPI Name	AVR Squid Cable Pin #	Itsy Pin Name
SCK	1	SCK
GND	2	G
MISO	3	MISO
VTG	4	3.3V
Reset	6	RST
MOSI	9	MOSI

Finding the Target Pins

- Each target pin corresponds to a specific pin on your Itsy.



Use the wiring kit to make connections between the Itsy and the Programmer.

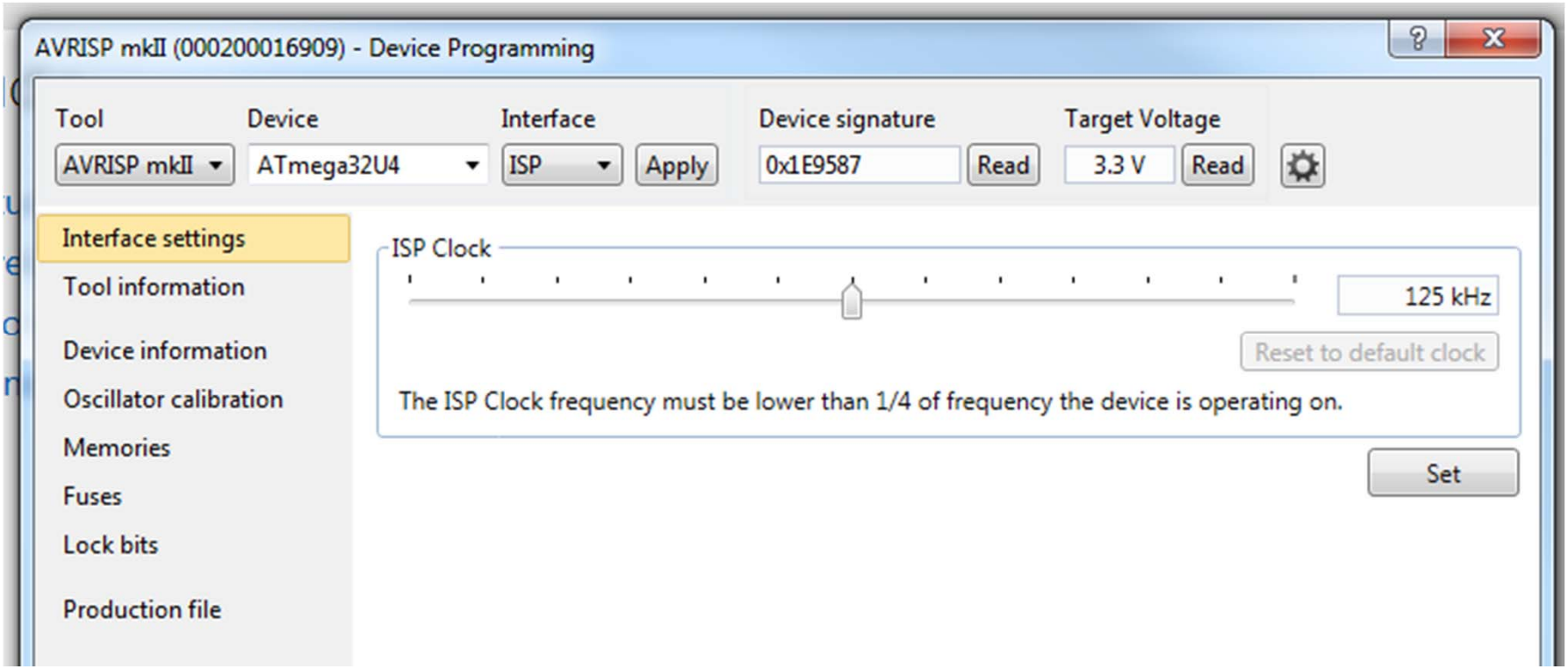
Do not yet plug in the programmer to your PC!

Connecting the Programmer

- Once you have wired up the programmer you must (1) plug in the itsy to your USB and (2) plug in the programmer to your USB

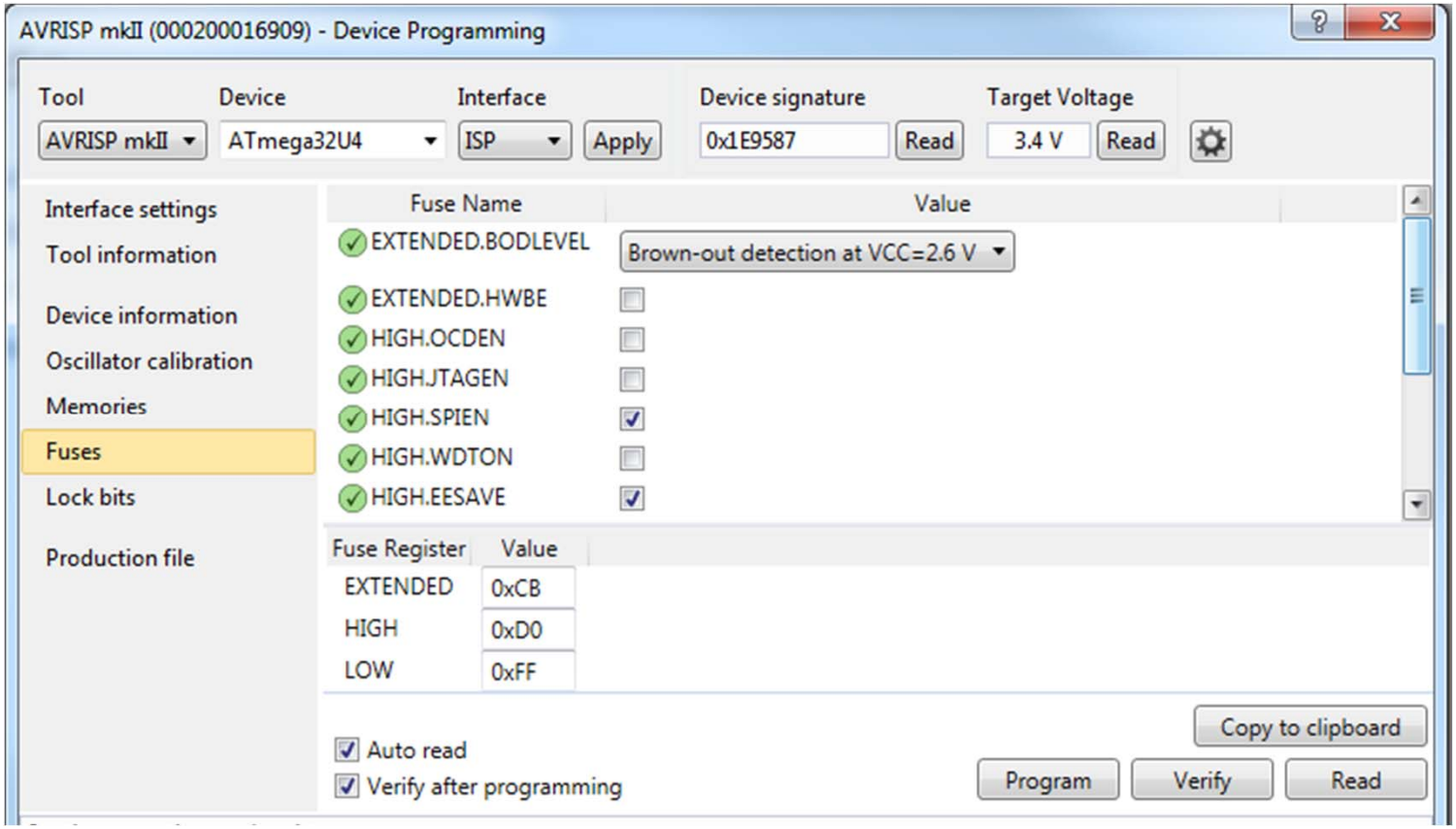
- Launch Atmel Studio and select the Programmer (Tools -> Device Programming)

Select the Programmer and Target



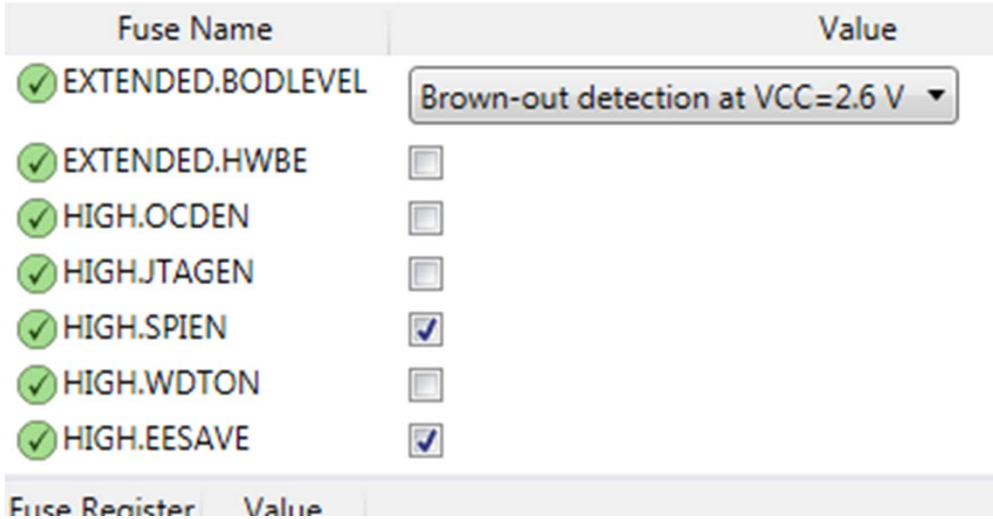
Clicking “apply” simply indicates if you can talk to programmer. If can read device signature and target voltage, then are likely communicating with device. AVRISP LED will turn green as well.

Enable/Disable Fuses



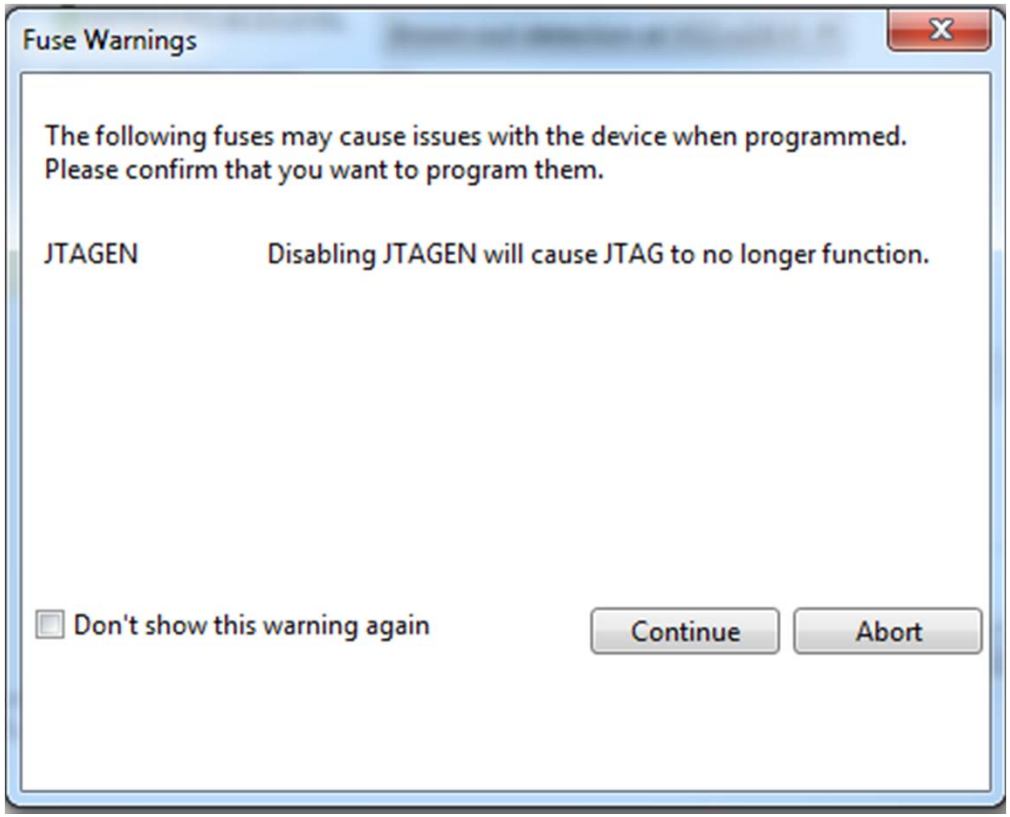
Regarding Fuses….

- Extremely ambiguous in ATMEL whether fuse is enable/disabled or blown/not blown.
- Which functionality is enabled if all the check boxes are green?
- Does checking a box enable or disable the functionality?



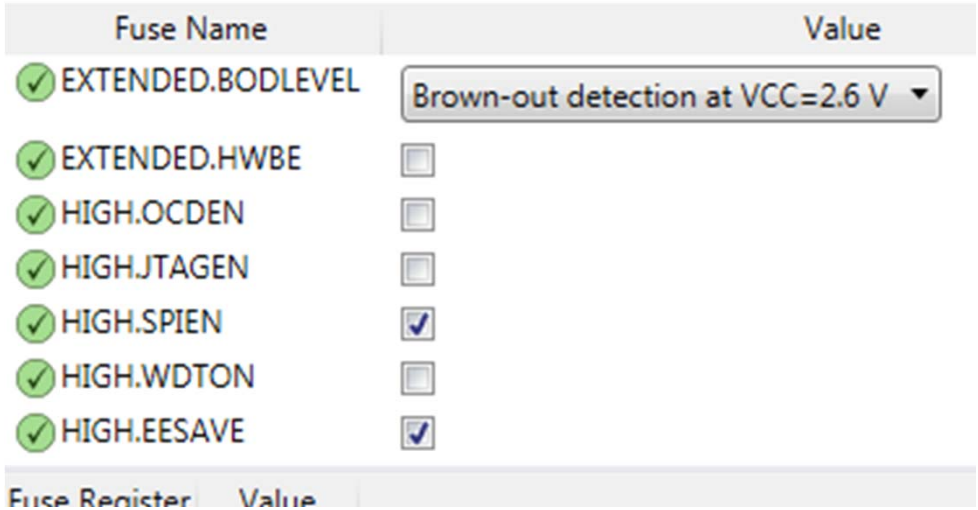
The warnings are even worse

- Check the JTAG box and you get this warning.
- Is JTAG being enabled or disabled?
- Warning does not indicate what you are going to do, just simply states you could screw up



General Fuse Rules

- Assume that checking box enabled that functionality
- Will need to change several fuses to enable JTAG. Will affect power measures.
- Never uncheck SPIEN!!!!

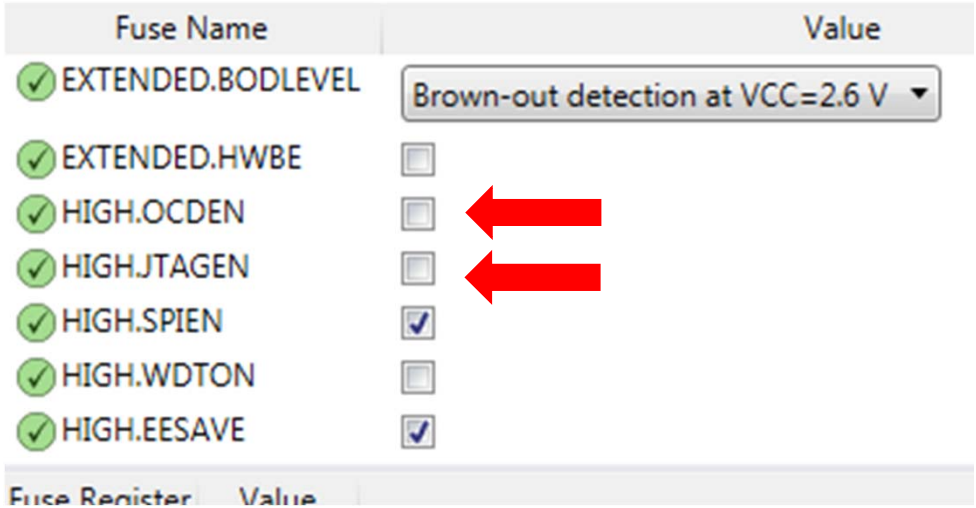


Never Uncheck SPIEN!!!

You will brick the microprocessor and have to purchase/re-solder a new one.

Enabling the JTAG Interface via SPI

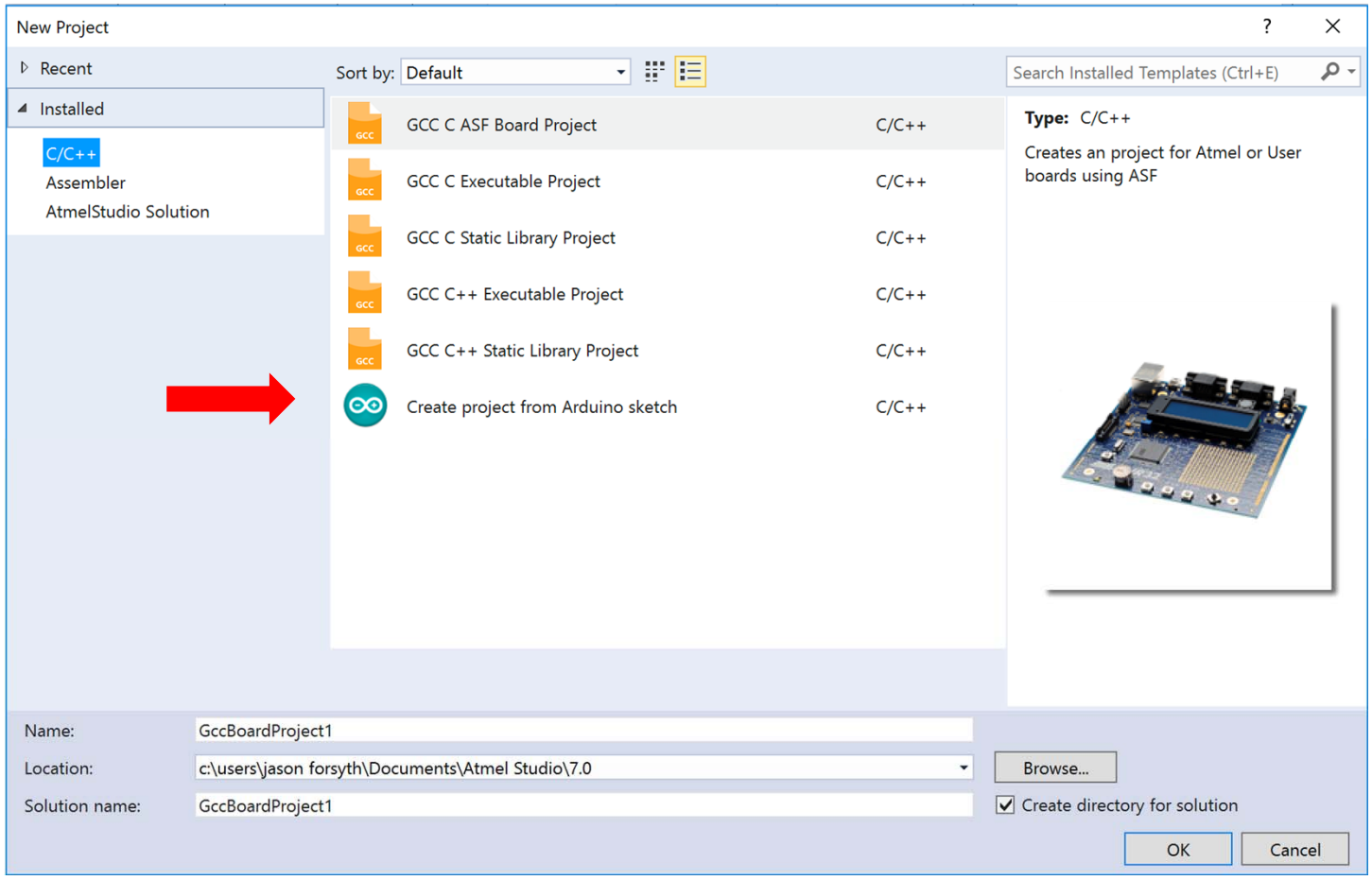
- Enable JTAGEN and OCDEN fuses via ISP connection. Then click Program.
- Now you can reset your connections to use the JTAG interface rather than SPI.



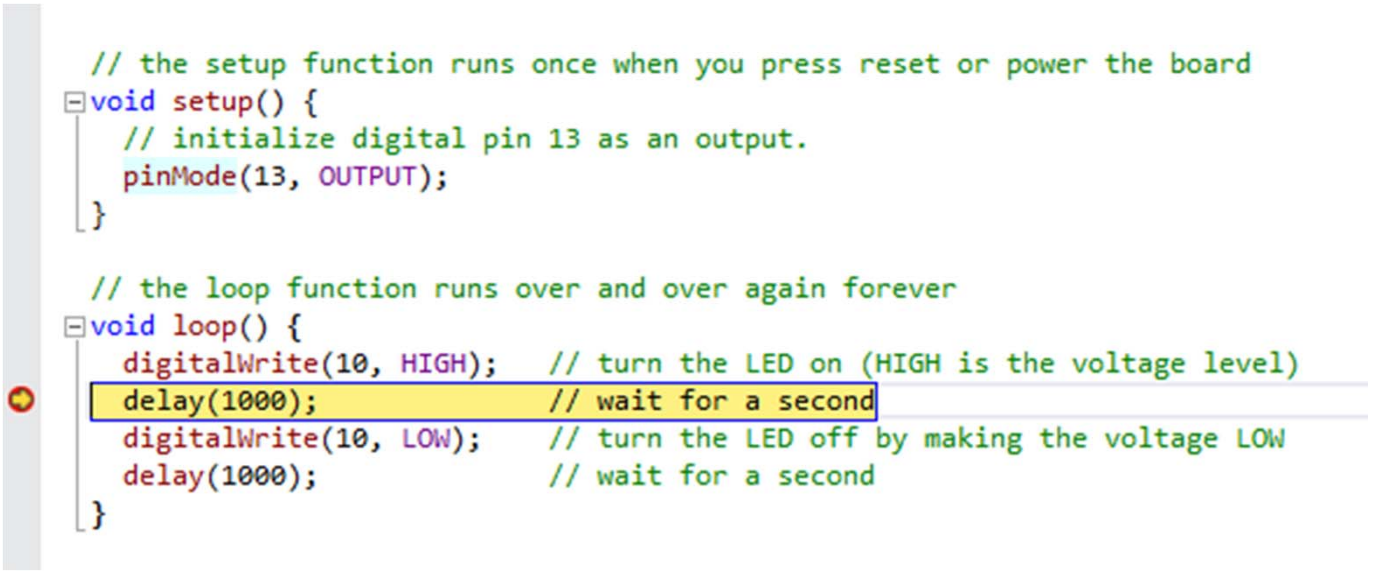
JTAG Pin Connections

JTAG Name	AVR Squid Cable Pin #	Itsy Pin Name
TCK	1	Analog 3 (A3)
TMS	5	Analog 2 (A2)
TDI	9	Analog 0 (A0)
TDO	3	Analog 1 (A1)
VTG	4	3.3V
GND	2,10 (must connect both)	Ground

Setting Up an “Arduino” Project in ATMEL Studio



Now we can debug programs directly on the board..

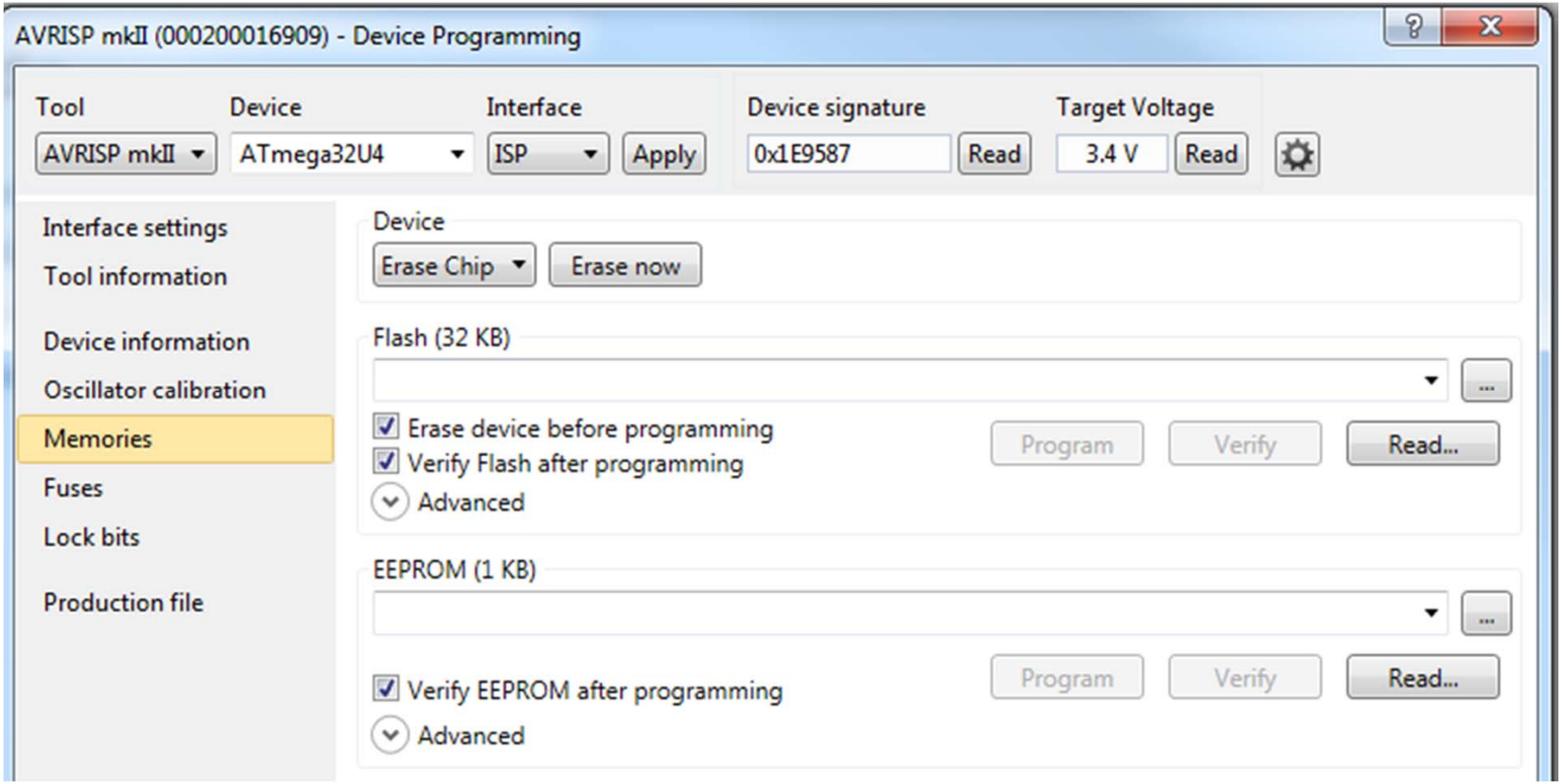


```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(10, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);              // wait for a second
  digitalWrite(10, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);              // wait for a second
}
```

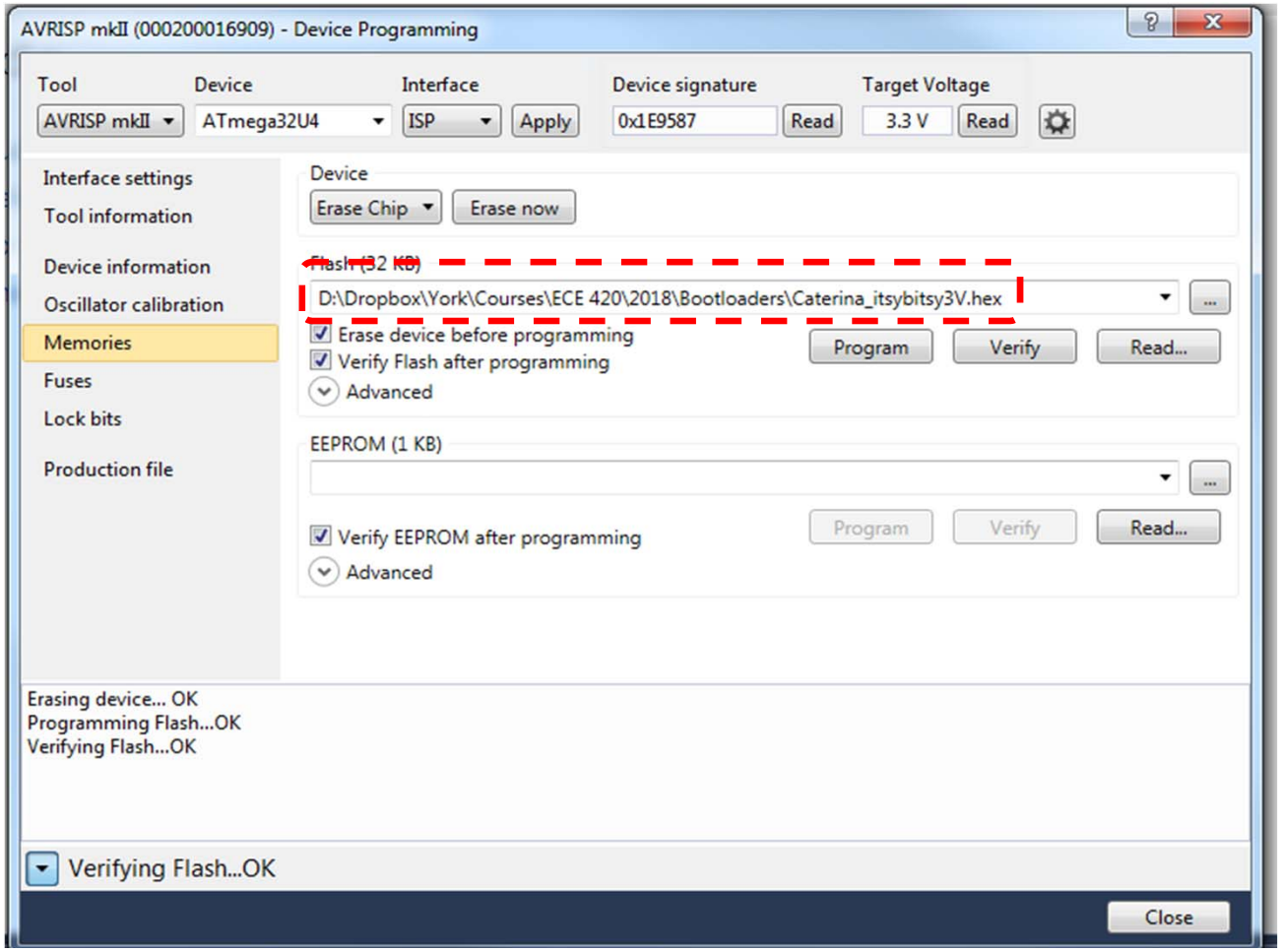
Assignment

- Use the SPI/ISP interface to access the 32u4 fuses. Turn on JTAG and OnChip Debug (check the boxes).
- Re-wire to connect to JTAG interface.
- Load example “Blink” program and set break point.
- Upload screen shot showing debugging Blink.
- (Optional): Restore Caterina bootloader to make accessible in Arduino

Addendum: Restoring the Boot Loader



Erase now deletes all memory. Can be restored by uploading binary .hex file.



If you want to use “Arduino” again, you need to reprogram with the Caterina 3.3V bootload. Available on Canvas.