

Engineering Innovation Week

University of Johannesburg (23 June – 27 June 2014)

Introduction and Getting Started



What will you make?



What Will You Make?

Intel® Galileo – Easily try out technology solution ideas



Intel® Edison - Create wearables and other products for the Internet of Things



Product Brief
Intel® Quark SoC X1000

Product Overview
The Intel® Quark SoC X1000 is the first product in a new roadmap of innovative, small core products targeted at rapidly growing areas ranging from industrial IoT to wearables. It will bring low power and Intel compute capabilities for thermally constrained, fanless, and headless applications. With its security and manageability features, this SoC is ideally suited for the Internet of Things (IoT) and for the next wave of cost-effective intelligent connected devices.

Featuring secure boot, extended lifecycle support, extended temperature and ECC, this processor offers an excellent solution for embedded market segments such as transportation, energy, and commercial and industrial control. The processor remains software compatible with previous 32-bit Intel® architecture and complementary silicon.

This Pentium® GA compatible, single-core, single-threaded SoC offers rich I/O capabilities and flexibility via high-bandwidth interfaces such as PCI Express® and USB 2.0 and offers the interfaces for a broad range of connectivity options such as two Ethernet® interfaces on-chip and interfaces to connect Cellular, Bluetooth®, ZigBee®, and other connectivity options. Using SDIO/SD/eMMC card interfaces, SPI, UART, and GPIO pins, the SoC connects seamlessly to sensors and various memory options.



IoT Software Stack
The Intel® Quark SoC X1000 is provided with a software suite of interoperable security, manageability and connectivity features, allowing true scalability in performance and features. It is supported by the Wind River Intelligent Device Platform (IDP) which is the operating system and middleware software stack that binds together the connectivity, security, and management on a scalable Intel architecture processing solution. This software solution is specifically packaged and validated to support the ability for developers to move immediately to the creation of high value applications and services.

Software Stack Highlights

- Provides interoperable connectivity, security, and manageability features on a scalable Intel architecture processing solution.
- Packaged to enable rapid application and service development.
- Validated and tested to assure seamless interoperability.
- Compatible and interoperable with the IoT stacks on other Intel processors, such as Intel® Atom® processor and Intel® Core™ processor.

Intel® Quark SoC X1000 Software Stack

Tools	WindRiver® IDP v2.0	Intel® Embedded control
	WindRiver® Linux® 5.0	WindWorks®
	GRUB (optional)	Open Source UEFI Bootloader
	Intel® Quark SoC X1000	Intel® 8, 3rd party I/O

Intel Facts

- Global headquarters: Santa Clara, California
- 107,600 employees worldwide (at end of 2013)
- 55 percent of employees reside in the U.S.
- 2013 net revenue ~ R560 billion

Intel's vision: This decade we will create and extend computing technology to connect and enrich the life of every person on earth

Team Introductions



Thabani Khupe



Trevor Cooper



Adrian Burns



Matt Royer



Erik Nyquist (not here)



D. John Oliver

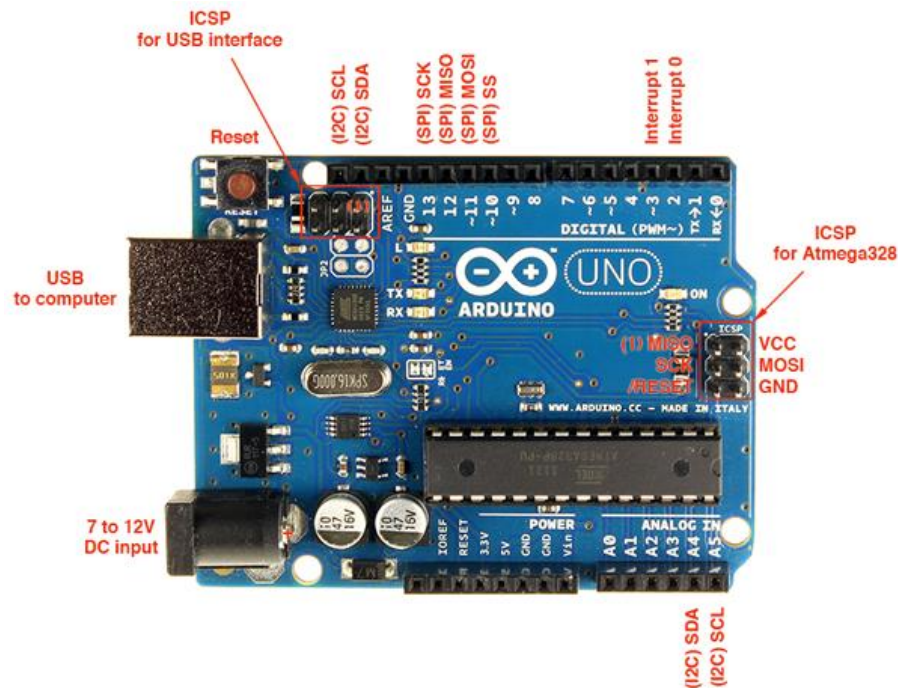
Schedule for the week

	Monday June 23	Tuesday June 24	Wednesday June 25	Thursday June 26	Friday June 27
8:00 AM	Arrive and mingle	Arrive and mingle	Arrive and mingle	Arrive and mingle	Arrive and mingle
8:30 AM	Intel/Team introduction		Visit Helen Joseph Hospital to see the challenge!		
9:00 AM	Review plan for the week		Review problem and brainstorm		
9:30 AM		Controlling devices (Lesson and projects)		Build	
10:00 AM					Project Show & Tell
10:30 AM	Getting started with Galileo				
11:00 AM			Build		
11:30 AM					
12:00 PM	Lunch	Lunch	Lunch	Lunch	
12:30 PM					
1:00 PM					
1:30 PM	Basic I/O (Lesson and projects)	Sensors (Lesson and projects)	Build	Build	
2:00 PM					Recognition & Celebration
2:30 PM					
3:00 PM	HS students join 3 - 6pm	HS students join 3 - 6pm	HS students join 3 - 6pm	HS students join 3 - 6pm	
3:30 PM					
4:00 PM					
4:30 PM		Connectivity (Lesson and projects)			
5:00 PM	Discussion / review				
5:30 PM					
6:00 PM	Entrepreneur Session	Entrepreneur Session	Entrepreneur Session	Entrepreneur Session	
6:30 PM					
7:00 PM	Wrap-up & review	Wrap-up & review	Wrap-up & review	Wrap-up & review	
7:30 PM					

What is Arduino?

An open-source hardware and software platform for building electronics projects

- A physical programmable circuit board (often referred to as a microcontroller)
- A piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board (supports Mac/Windows/Linux)



```
Blink | Arduino 1.0.3

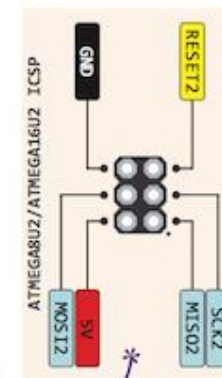
Blink
Turns on an LED on for one second, then off for one second, repeatedly.

This example code is in the public domain.
*/


// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;


// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

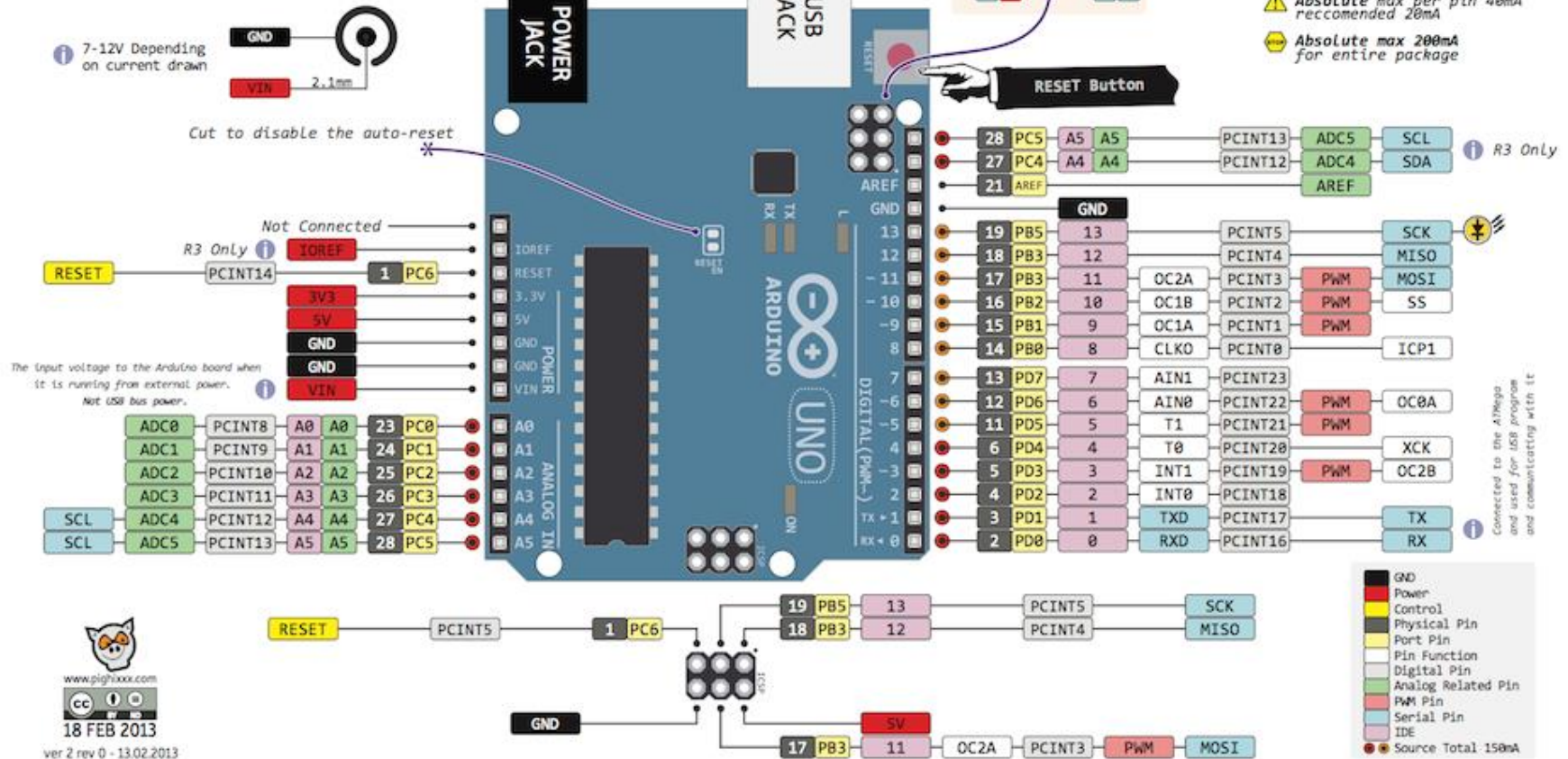
// the loop routine runs over and over again forever:
void loop(){
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);             // wait for a second
  digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW
  delay(1000);             // wait for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

THE
DEFINITIVE
ARDUINO
UNO
PINOUT DIAGRAM

 **Absolute max per pin 40mA**
recommended 20mA

 **Absolute max 200mA**
for entire package



10 Great things about Galileo

Shield Compatibility

WiFi/Ethernet Library Compatibility

Familiar Arduino IDE

Real Time Clock

MicroSD Support

USB Host Port

Serial Connectivity

Linux on Board

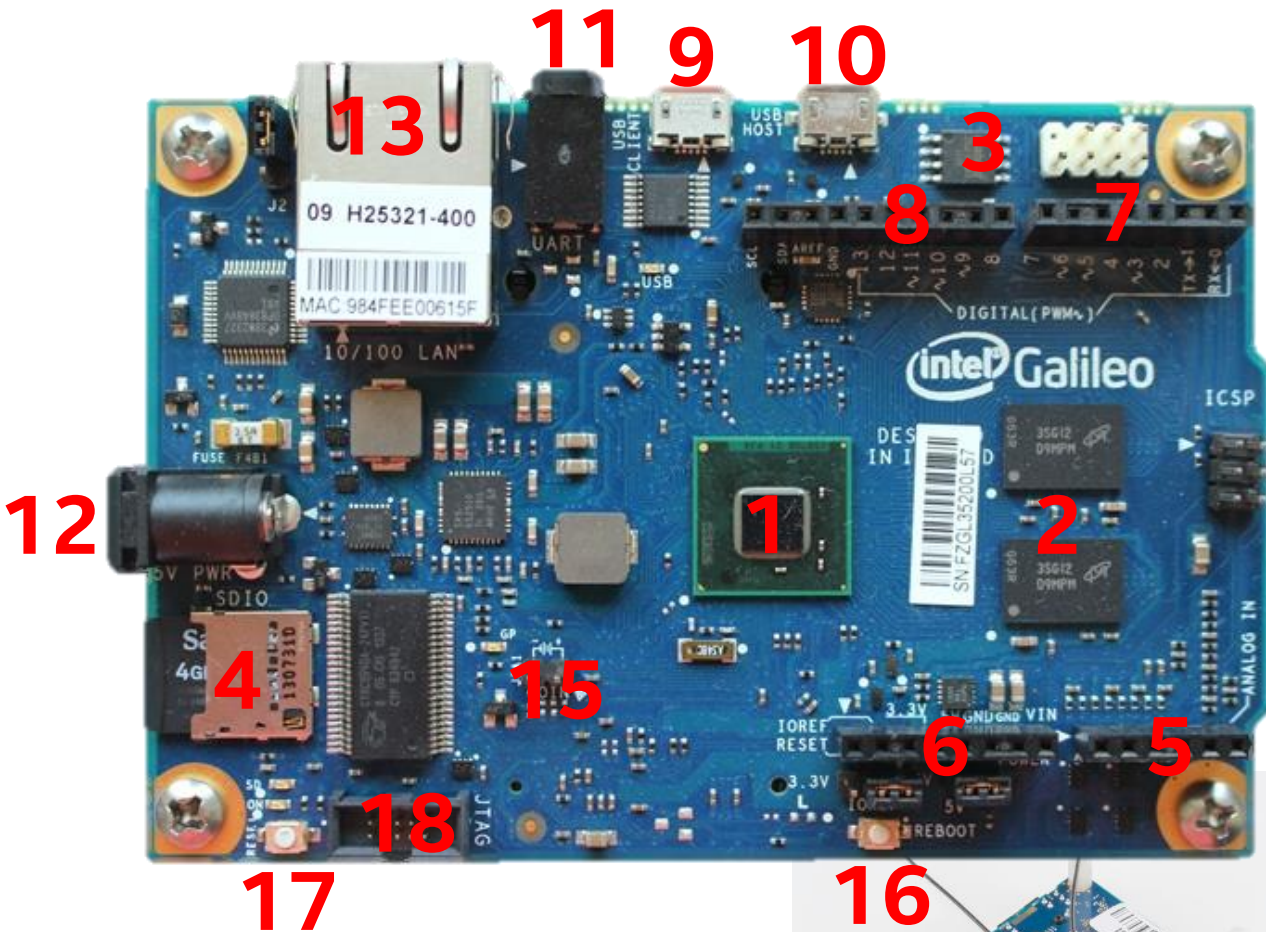
Works with PCI

TWI/I2C, SPI Support



Source Makezine: <http://makezine.com/2013/10/03/10-great-intel-galileo-features/>

Galileo Board Overview



1. Processor (Quark X1000)
2. RAM (256 MB DRAM)
3. Flash Memory
4. 32GB Micro-SD Card Slot
5. 6-pin analog input header
6. 8-pin power header
7. 8-pin digital I/O
8. 10-pin digital I/O
9. USB Client Port
10. USB Host Port
11. Serial Port (RS-232)
12. Power Input (5V)
13. Ethernet Port
14. Mini PCI Express Slot
15. Clock Battery Power
16. Reboot Button
17. Reset Button
18. JTAG Header

Galileo - Arduino Hardware Compatibility

Standard Arduino UNO connectors:

- Supports 3.3V and 5V **shields**
- 8-pin **power** header (3.3V, 5V, GND, Reset, etc.)
- 6-pin **analog input** header (A0-A6)
- 8-pin **digital** I/O header (D0-D7), which includes **UART** on pins 0/1, **PWM** on pins 3, 5, and 6
- 10-pin **digital** I/O header (D8-SCL), which includes **I²C** pins and **PWM** on pins 9, 10, and 11
- 2x3-pin ICSP header breaks out **SPI** pins

Galileo - Capabilities beyond Arduino UNO

Hardware

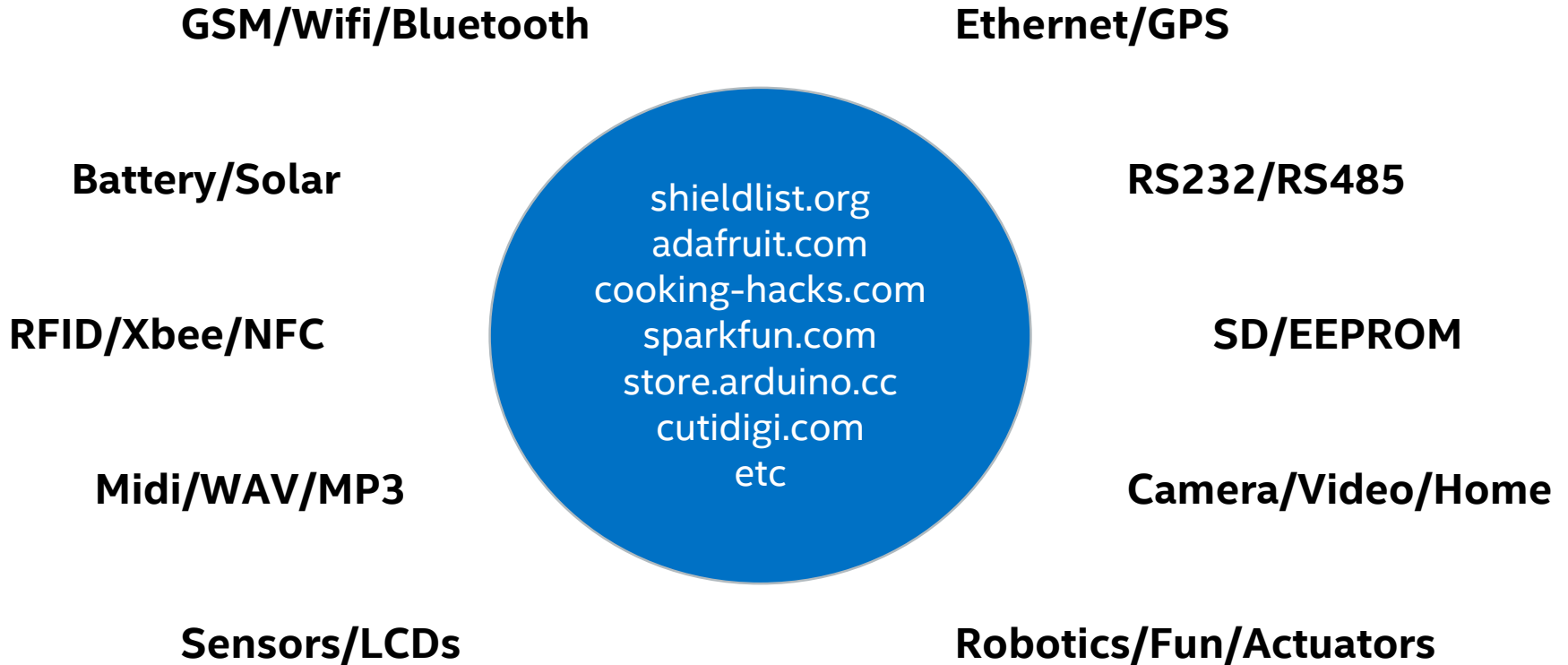
- 400MHz Quark SoC
- 512KB SRAM
- 256MB DRAM
- 8 MByte Flash
- Micro-SD slot
- Mini-PCI Express slot
- 100Mb Ethernet port
- USB Host port
- USB Client Port

Software

- Linux OS on Board
- Intel SW Tools
 - IoTDevKit
 - JTAG debugger (OpenOCD)
 - Intel Software Tools (ISS)

Arduino Shields

There are **hundreds** of shields available from dozens of suppliers



Galileo Shield

- Supports 3.3V and 5V shields
- Hardware compatible with most existing shields
- HW/SW reuse reuse reuse (don't reinvent the wheel)!

intel Galileo

libelium 



 e-Health



<http://arduino.cc/en/Main/ArduinoMotorShieldR3>



<http://www.seeedstudio.com/depot/Solar-Charger-Shield-V2-p-914.html>

Galileo Support

<http://www.intel.com/content/www/us/en/do-it-yourself/galileo-maker-quark-board.html>



Galileo support site: <http://www.intel.com/support/galileo/>

- How to videos: <http://www.intel.com/support/galileo/howtovideos.htm>

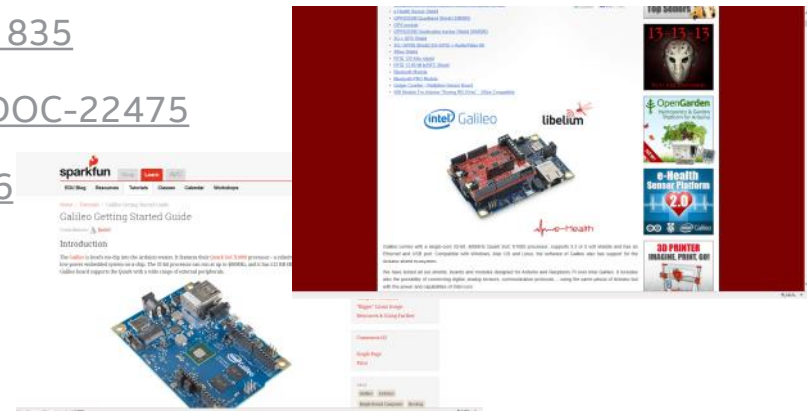
Intel Maker community: <https://communities.intel.com/community/makers>

- Documents: <https://communities.intel.com/community/makers/documentation/galileodocuments>
- Data sheet: <https://communities.intel.com/docs/DOC-21835>
- Board User Guide: <https://communities.intel.com/docs/DOC-22475>
- Drivers: <https://communities.intel.com/docs/DOC-22226>

There are many Galileo communities e.g.

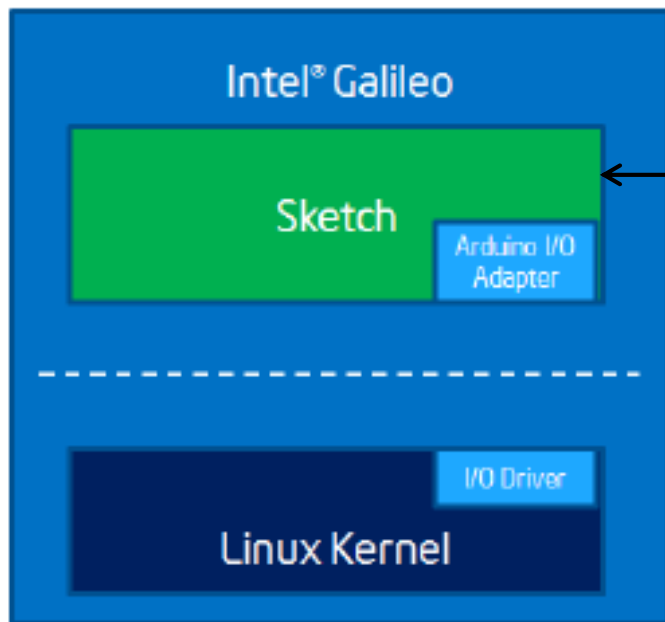
Sparkfun getting started guide: <https://learn.sparkfun.com/tutorials/galileo-getting-started-guide>

Using Galileo with Arduino shields: <http://www.cooking-hacks.com/documentation/tutorials/intel-galileo-tutorial-using-arduino-and-raspberry-pi-shields-modules-boards>



Arduino Software Compatibility

Familiar Arduino IDE with example sketches



**Load Sketch from PC
to Galileo**

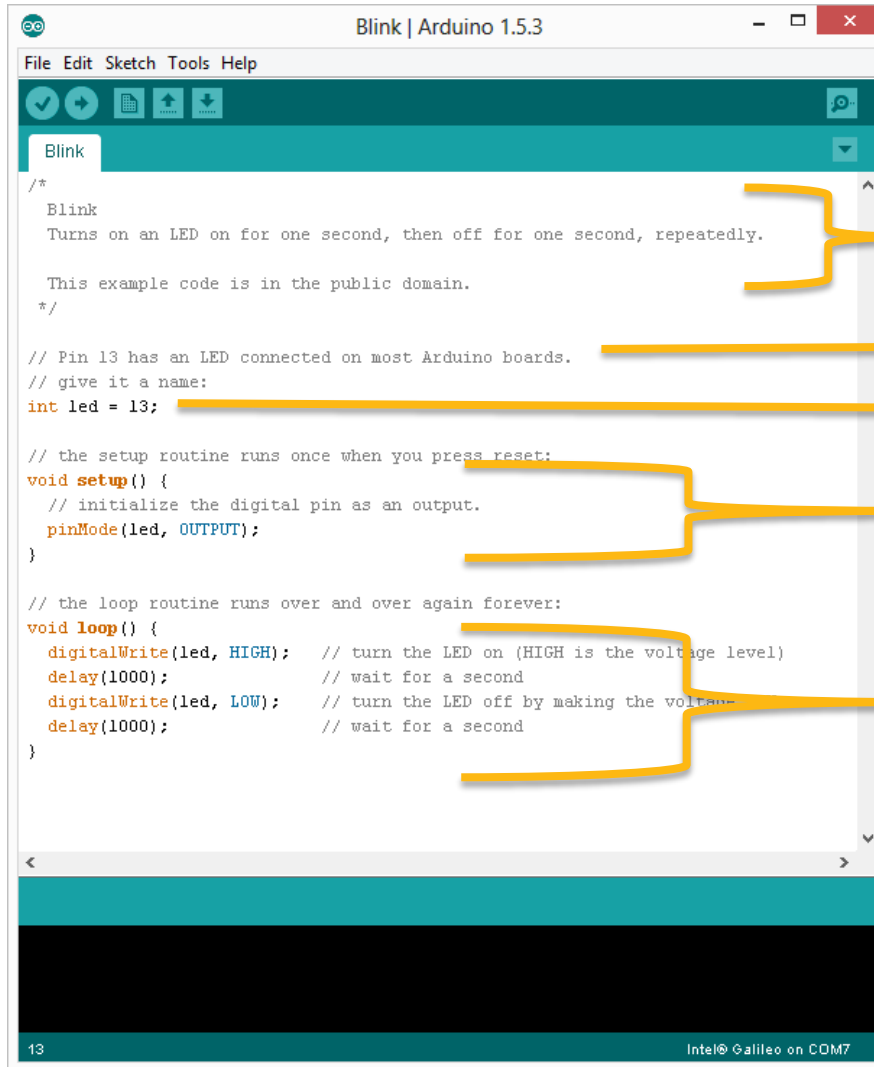
```
/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeatedly.
 *
 * This example code is in the public domain.
 */
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

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

24 Arduino Mega (ATmega1280) on /dev/tty.usbserial-A600enbz

Sketch Overview



The screenshot shows the Arduino IDE interface with the 'Blink' sketch loaded. The code is as follows:

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
  */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

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

Annotations with yellow lines point to specific parts of the code:

- Comment Block:** Points to the multi-line comment at the top of the sketch.
- Comment:** Points to the single-line comment `// Pin 13 has an LED connected on most Arduino boards.`
- Global Variable:** Points to the line `int led = 13;`.
- Code executed once on initial load:** Points to the `void setup() { ... }` block.
- Code Loop ran continuously:** Points to the `void loop() { ... }` block.

Comment Block

Comment

Global Variable

Code executed once on initial load

Code Loop ran continuously

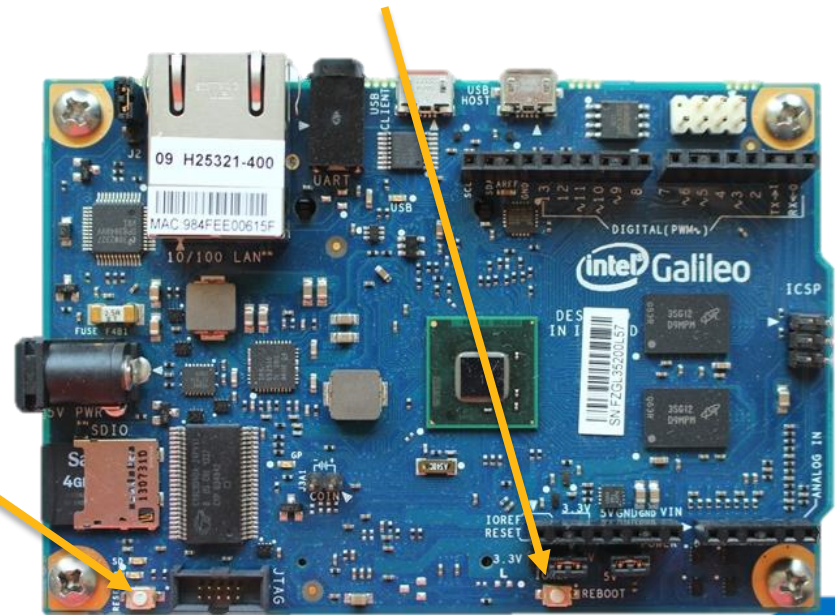
Reboot and Reset Button Differences

Reset Button

- Restart the Arduino sketch running on the Galileo. Includes re-executing the code defined within the Setup section
- Acts much more like the reset button you may be used to

Reboot Button

- Will reboot the entire Galileo - Linux included
- Boot time is about 30 seconds, so don't press this accidentally!



Lets Get Started!

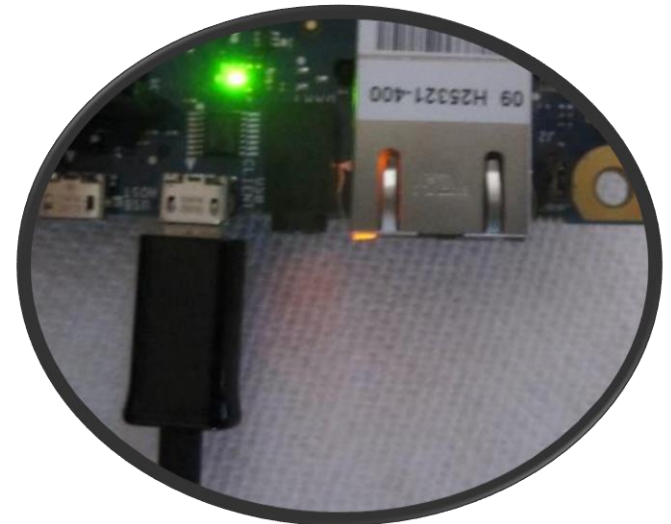
Arduino Environment

1. Download and install the latest Galileo IDE

- <https://communities.intel.com/docs/DOC-22226>
- If you are updating your IDE uninstall the previous IDE version first
- Download the zip file for your OS (includes the latest firmware to update your board): Intel_Galileo_Arduino_SW_1.5.3_on_Windows_v1.0.0.zip
- Extract into the C:\ ... use an unzip tool that supports an extended file path ... <http://www.7-zip.org/>

2. Connect the board using USB

- **WARNING:** Always connect the 5V power before any other connection
- Connect micro-B USB Client Port (closest to the Ethernet) to a PC
- Wait for Windows to begin its driver installation process



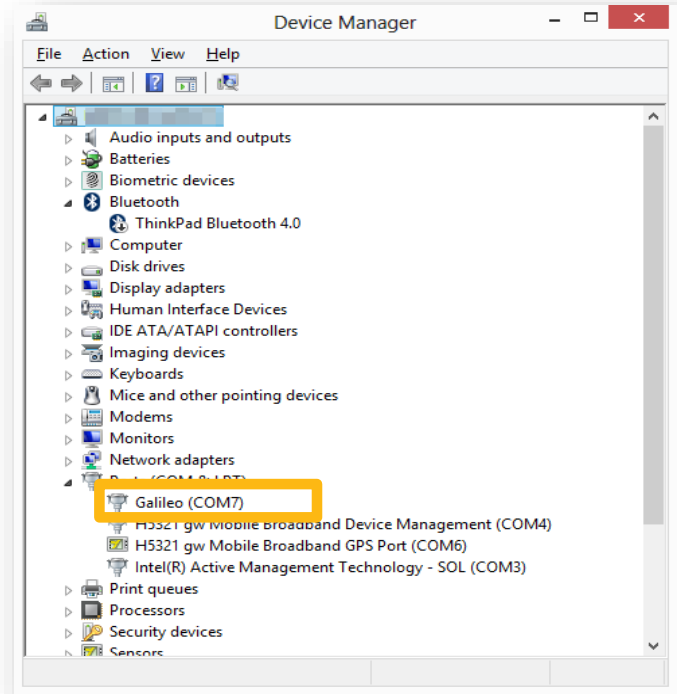
Arduino Environment

3. Install the drivers

- C:\arduino-1.5.3\hardware\arduino\x86\tools (correct driver file is linux-cdc-acm.inf)
- Once the driver is installed, Device Manager will show a Galileo (COMx) device under Ports (COM & LPT)

4. Launch the Arduino IDE application

- In the folder arduino-1.5.3, double-click arduino.exe
- Select the board via Tools -> Board -> Intel® Galileo
- Select the correct serial port using Tools -> Serial Port



Running your First Sketch – Blink LED

```
/*  
Blink  
Turns on an LED on for one second, then off for one second, repeatedly.
```

This example code is in the public domain.

```
*/
```

```
// Pin 13 has an LED connected on most Arduino boards.
```

```
// give it a name:
```

```
int led = 13;
```

```
// the setup routine runs once when you press reset:
```

```
void setup() {
```

```
    // initialize the digital pin as an output.
```

```
    pinMode(led, OUTPUT);
```

```
}
```

```
// the loop routine runs over and over again forever:
```

```
void loop() {
```

```
    digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
```

```
    delay(1000);             // wait for a second
```

```
    digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW
```

```
    delay(1000);             // wait for a second
```

```
}
```

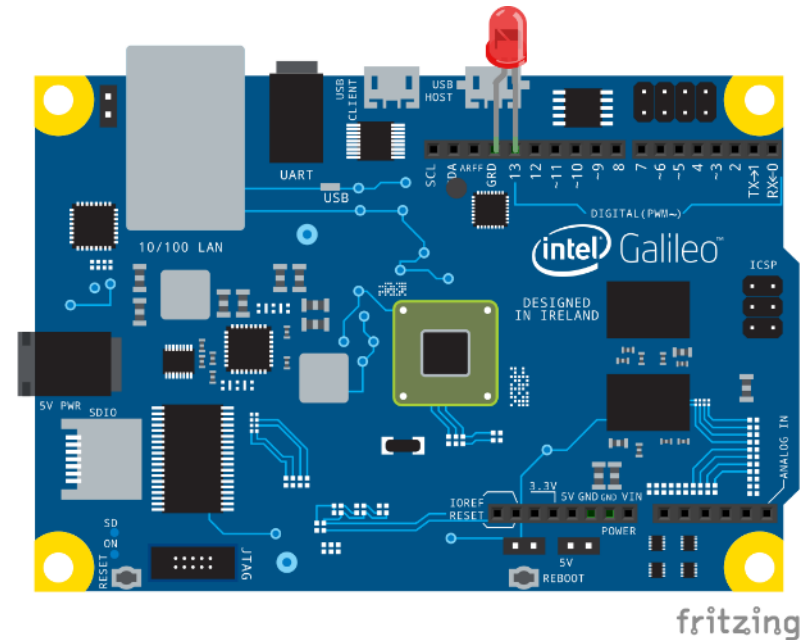


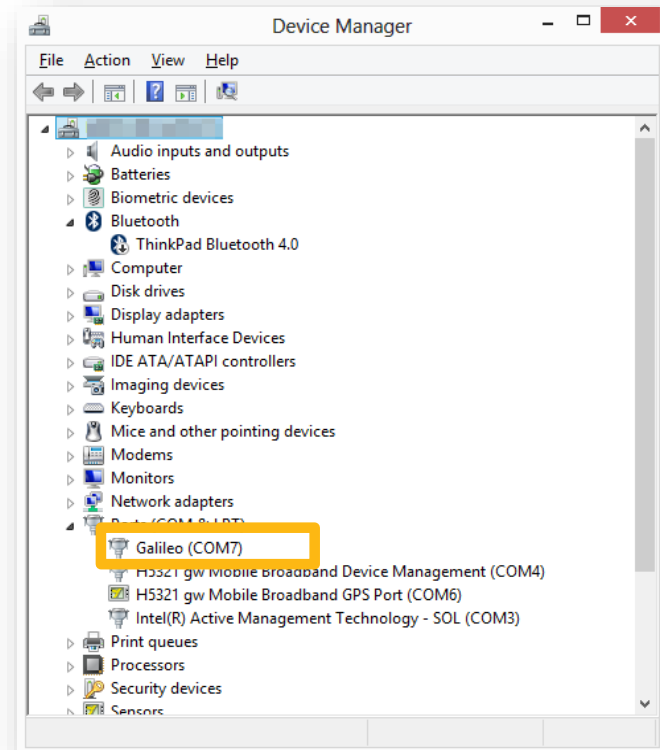
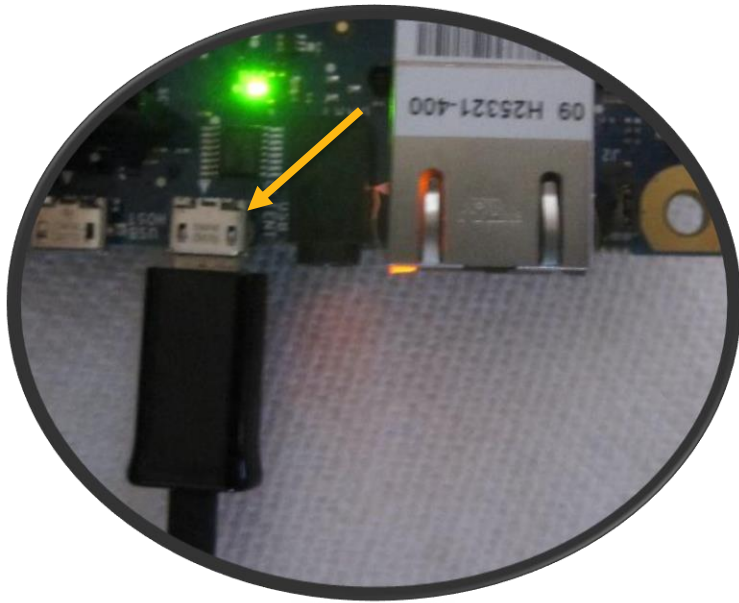
Image Source from Fritzing*

* Other names and brands may be claimed as the property of others

Project Files - USB Drive:\Lessons\Lesson2-BasicIO\Section1-DigitalWrite\Blink

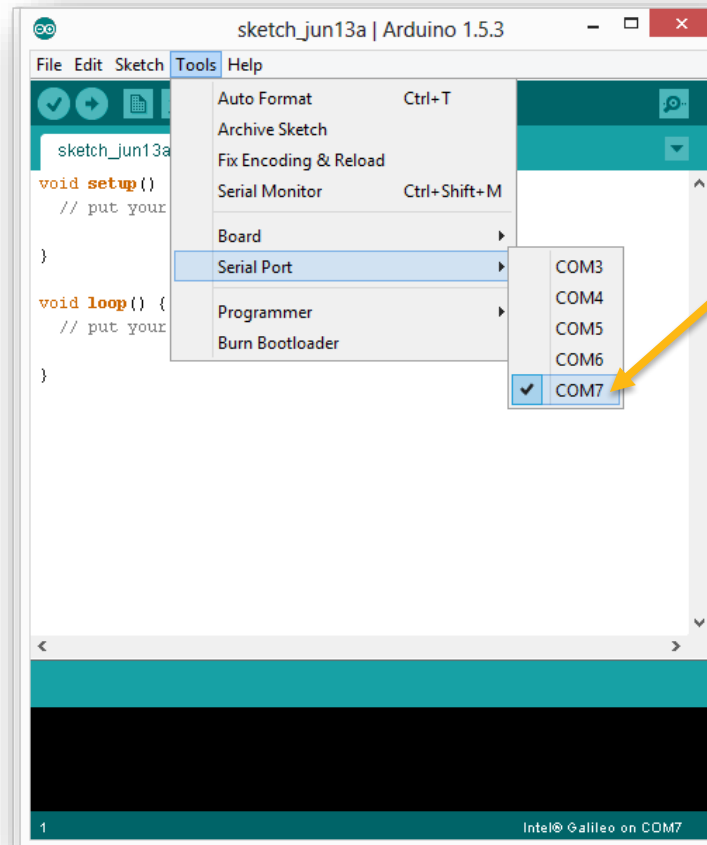
Uploading Sketch

1. Ensure Galileo 5v Power is connected first
2. Connect USB Cable to Galileo Micro-B (labeled USB Client) and Computer
3. Identify which COM port is assigned to Galileo Connection



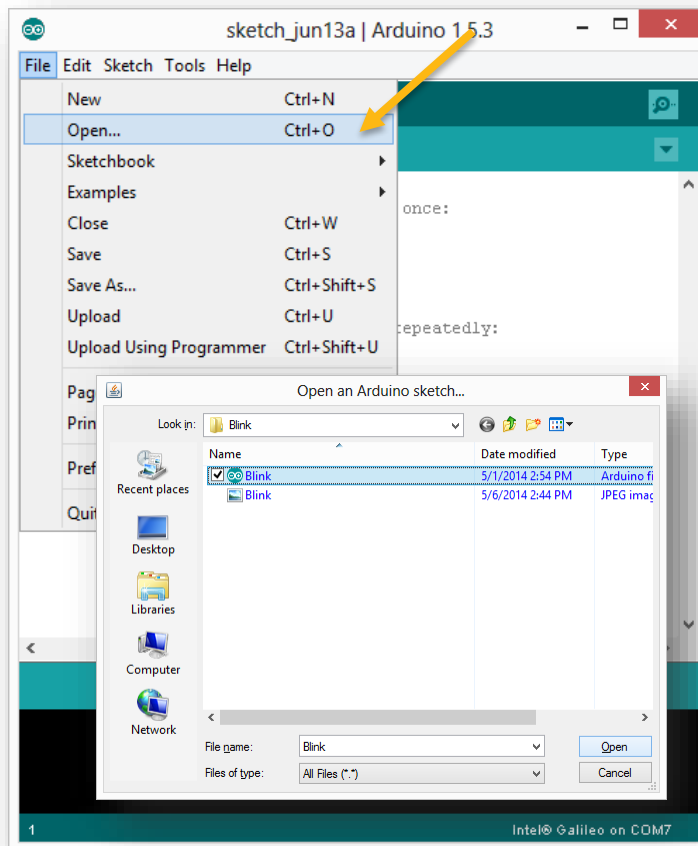
Uploading Sketch

3. Ensure proper COM port is selected within Arduino IDE, “Tools” -> “Serial Port” -> COMx

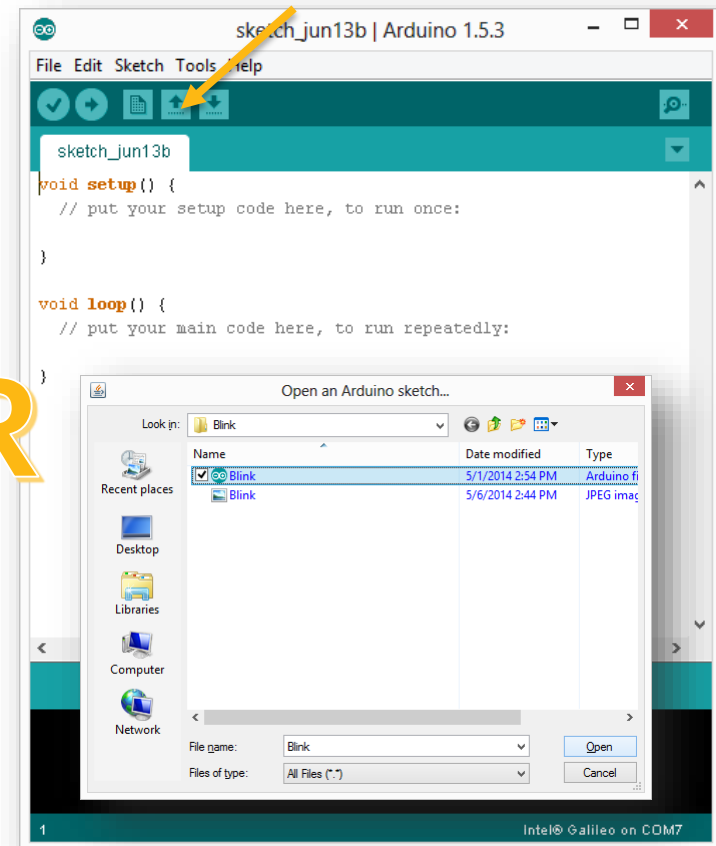


Uploading Sketch

4. If opening an existing sketch, Use “File” -> “Open”, Browse to desired sketch, and select “Open” or select “Open” icon

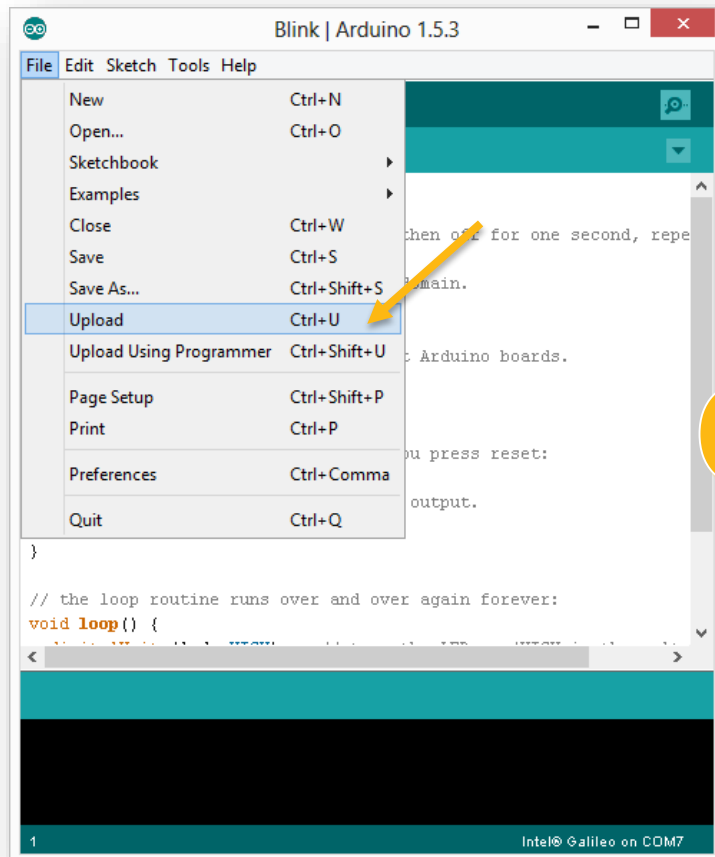


OR

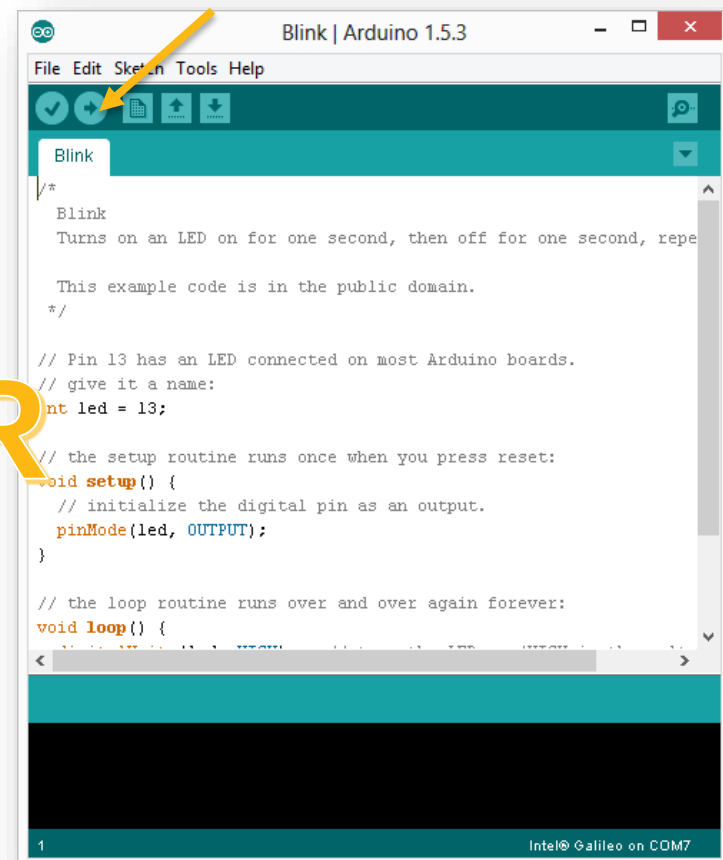


Uploading Sketch

- When ready to upload a sketch, select “File” -> “Upload” or select “Upload” icon

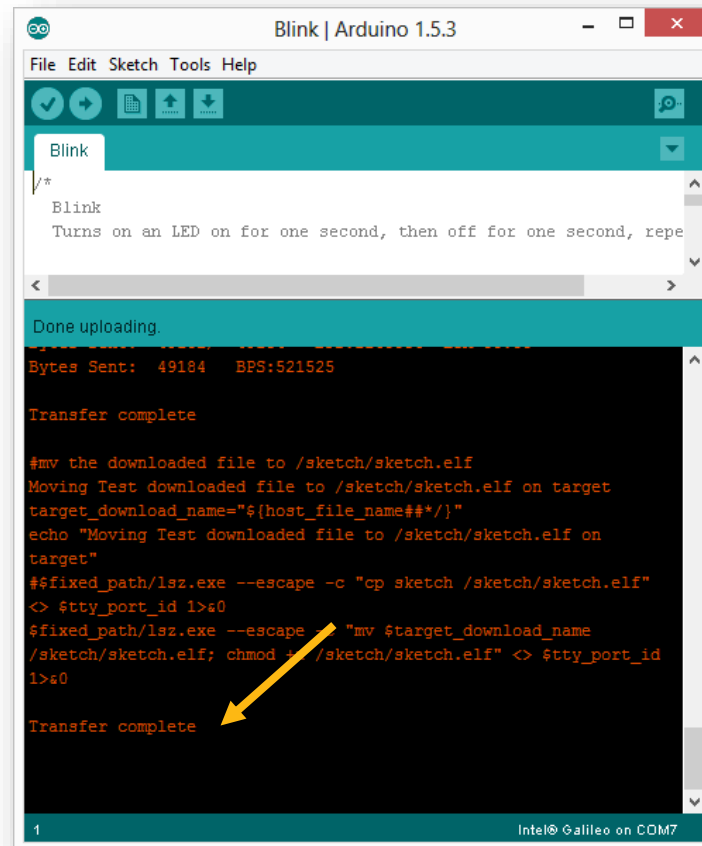


OR



Uploading Sketch

6. Ensure Sketch successfully loaded



The screenshot shows the Arduino IDE interface with the 'Blink' sketch selected. The terminal window displays the following output:

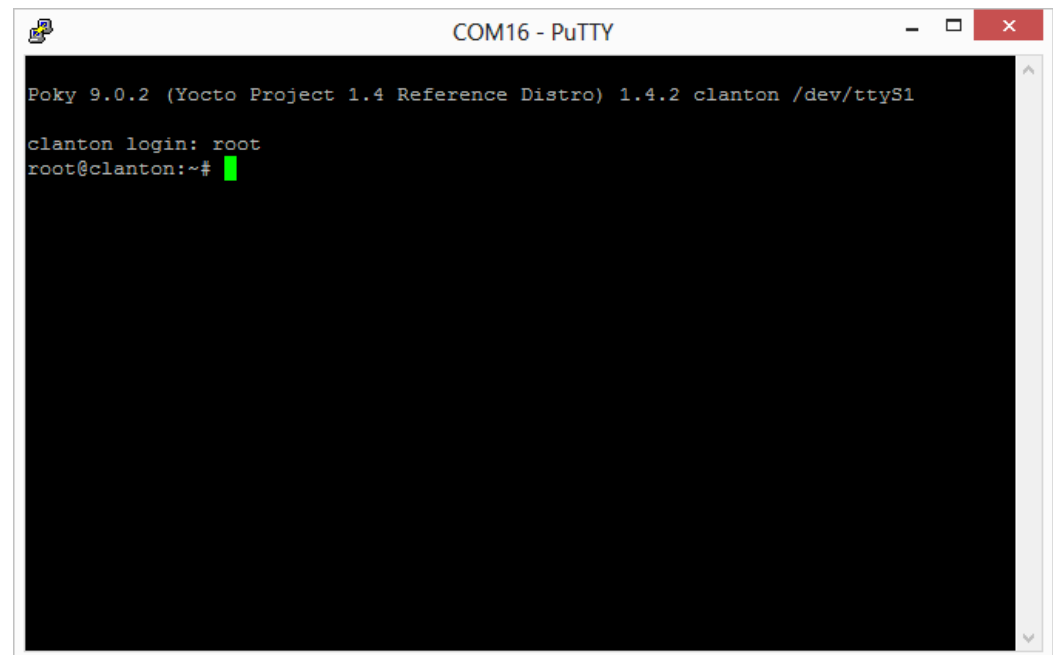
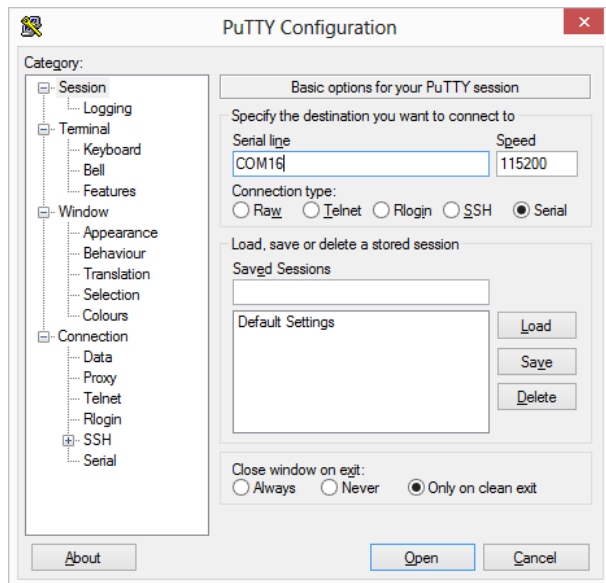
```
Done uploading.  
Bytes Sent: 49184 BPS:521525  
Transfer complete  
  
#mv the downloaded file to /sketch/sketch.elf  
Moving Test downloaded file to /sketch/sketch.elf on target  
target_download_name="${host_file_name##*/}"  
echo "Moving Test downloaded file to /sketch/sketch.elf on  
target"  
$fixed_path/lsz.exe --escape -c "cp sketch /sketch/sketch.elf"  
<> $tty_port_id 1>%0  
$fixed_path/lsz.exe --escape "mv $target_download_name  
/sketch/sketch.elf; chmod +x /sketch/sketch.elf" <> $tty_port_id  
1>%0  
Transfer complete
```

A yellow arrow points to the 'Transfer complete' message in the terminal.

The Terminal, interact with the Galileo's Linux half

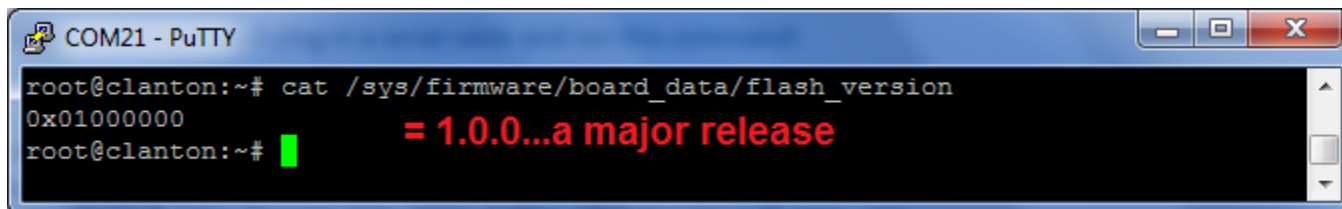
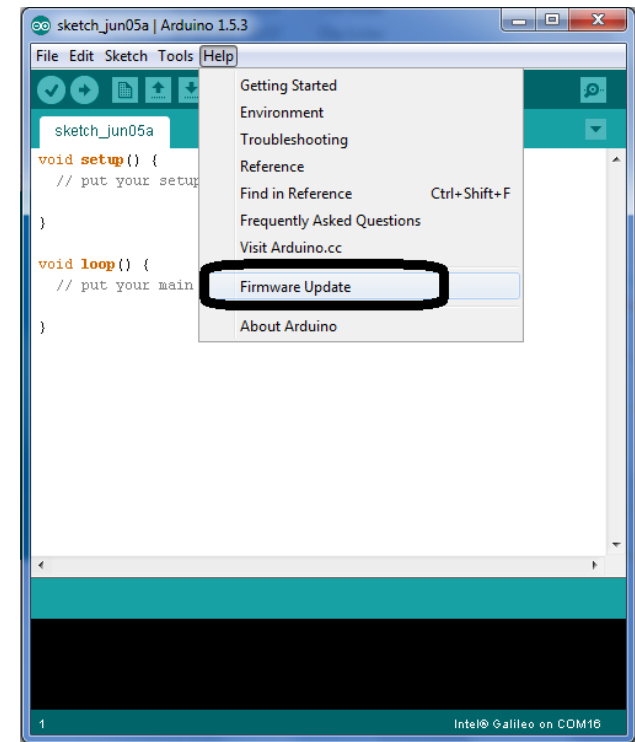
Allows you to watch / interact with Linux OS through the Serial Interface ... e.g. walk through the Galileo board boot

- Serial Terminal Emulator Program
 - Use e.g. Tera Term or (we will use) PuTTY
 - Set the Serial Port number (COMx) and change the baud rate to 115200 bps
 - Clanton Login: root



Firmware Upgrade (1.0.0)

- Remove all power from the board (USB and 5 V power cord)
- Remove the SD card from the board (if it is inserted)
- Power up the board by plugging in the 5V power supply
- Connect the USB cable to the USB Client Port
- Takes about 6 minutes and several popup messages (with no access to the IDE)
- To check the firmware version plug in a serial cable and run this command

A screenshot of a PuTTY terminal window titled 'COM21 - PuTTY'. The terminal shows the command 'cat /sys/firmware/board_data/flash_version' being executed. The output is '0x01000000', followed by a red text overlay that reads '= 1.0.0...a major release'. The prompt 'root@clanton:~#' is visible at the start of each line.

LINUX IMAGE for Intel Galileo on μ SD card

Overview

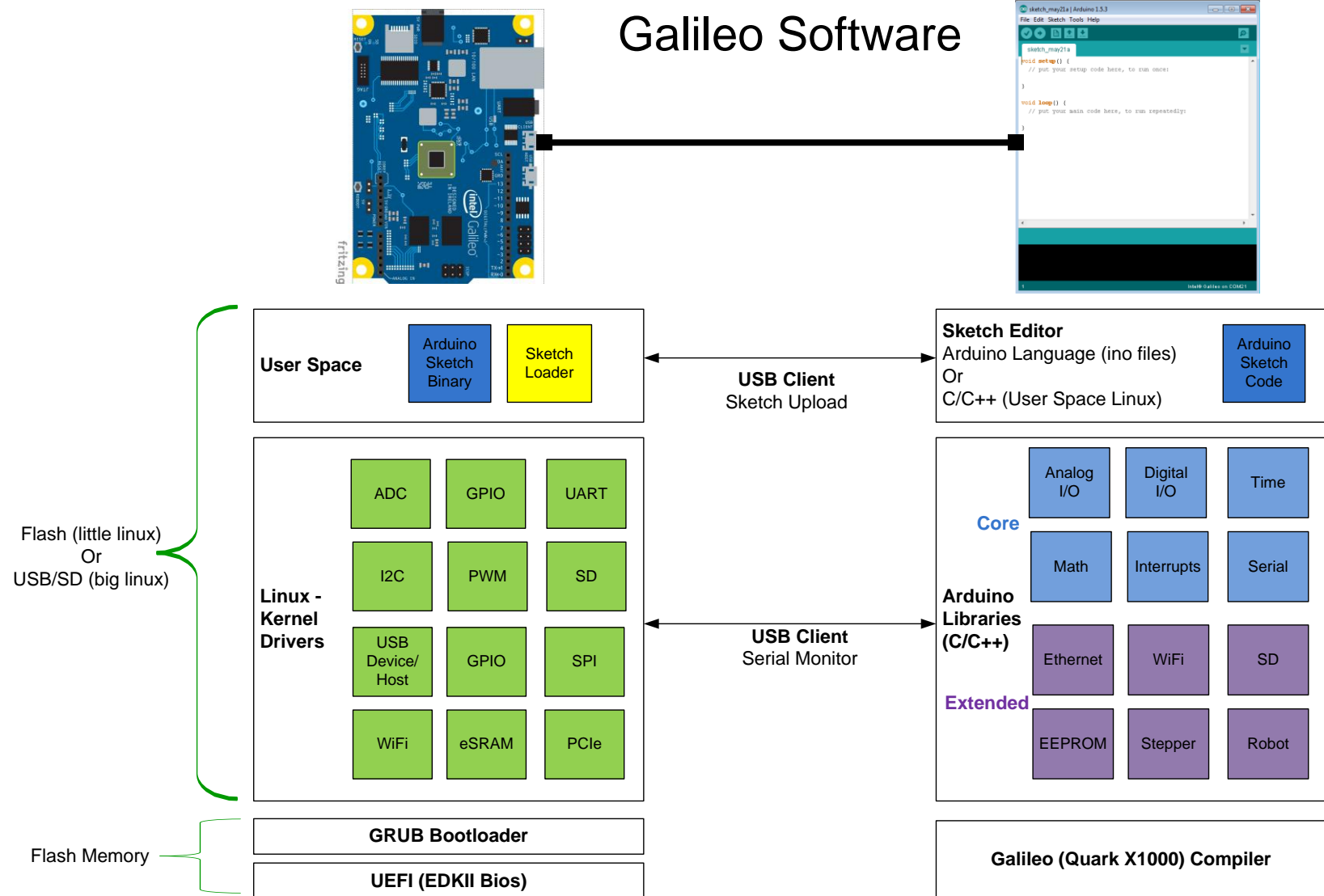
- Provides an alternative boot environment from the SPI Flash Memory
- Due to space constraint on Flash, SD image is required to use WiFi
- Additional Software and Drivers can be added to SD Boot image to enhance capabilities

Steps to Create

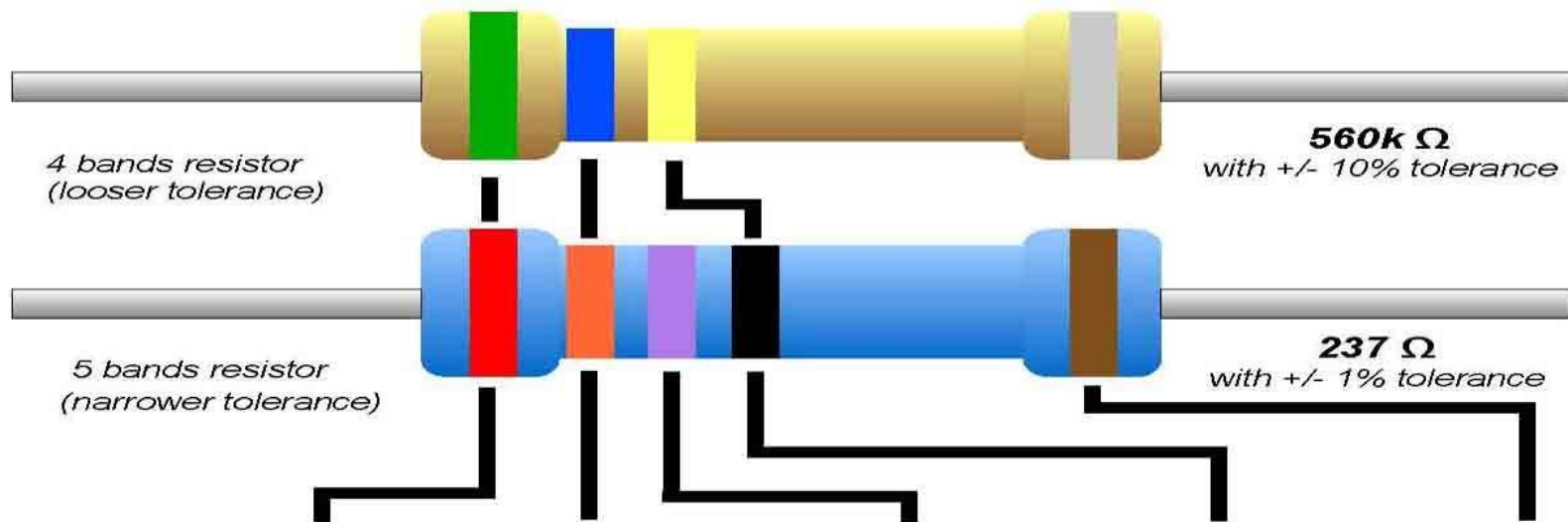
- Download LINUX_IMAGE_FOR_SD_Intel_Galileo_v1.0.0.zip file from <https://communities.intel.com/docs/DOC-22226>
- Format 32GB (or smaller) SD card as FAT or FAT32 (SDXC format is not supported)
- Copy all files and directories from the zip file to (top level) of your SD card
- Insert the SD card, then power on the board (first time you boot may take several minutes)

BACKUP

Software Stack



Resistor Color Code



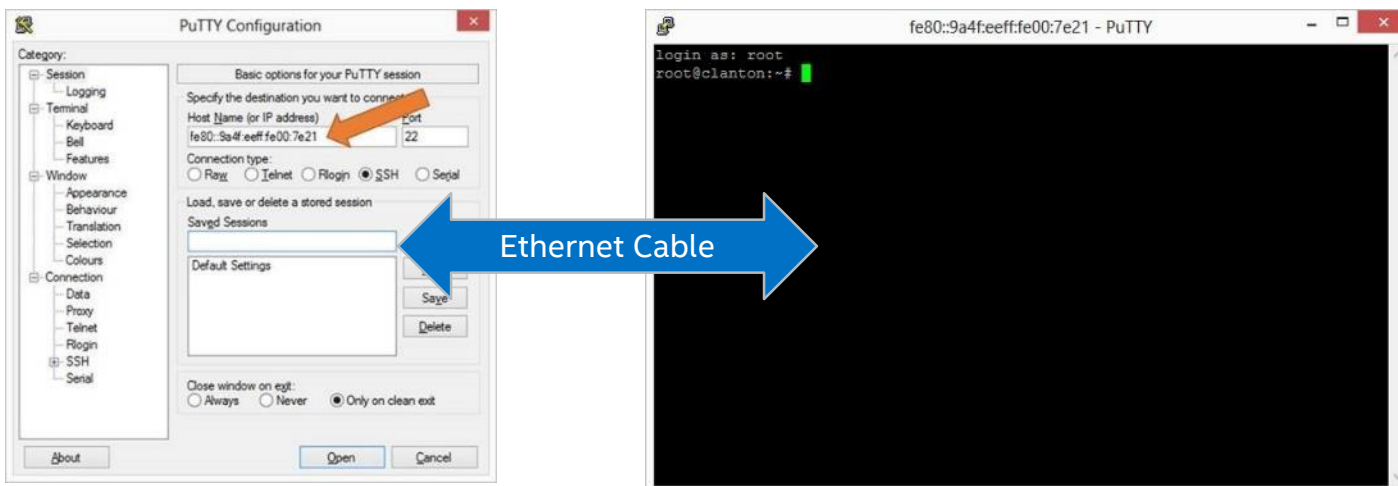
Color	1 st Band	2 nd Band	3 rd Band	Multiplier	Tolerance
Black	0	0	0	x 1 Ω	
Brown	1	1	1	x 10 Ω	+/- 1%
Red	2	2	2	x 100 Ω	+/- 2%
Orange	3	3	3	x 1K Ω	
Yellow	4	4	4	x 10K Ω	
Green	5	5	5	x 100K Ω	+/- 5%
Blue	6	6	6	x 1M Ω	+/- .25%
Violet	7	7	7	x 10M Ω	+/- .1%
Grey	8	8	8		+/- .05%
White	9	9	9		
Gold				x .1 Ω	+/- 5%
Silver				x .01 Ω	+/- 10%

Communicating with Galileo using Ethernet

If Galileo is already connected and accessible on the network through WiFi or Ethernet, just SSH straight to the Galileo network address

You can also connect the Galileo and Computer directly together with an Ethernet cable

- Ethernet Cable will auto Cross over
- Use instruction here <https://communities.intel.com/thread/45455> to connect

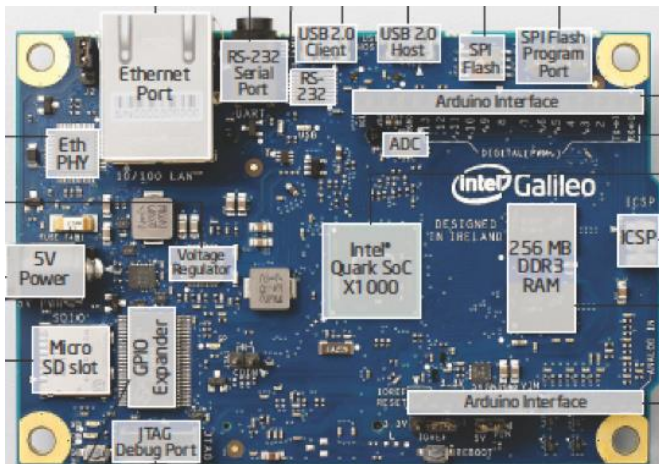


Knowledge Assumptions

- Computer concepts
 - CPU
 - Memory
 - I/O (interfaces)
 - Operating System
 - Machine code
 - Analog vs Digital
- Basic coding concepts
 - Variables and constants
 - Read/write concepts
- Compiling and executing
- Circuits
 - Electric current flow
 - Conductors / insulators
 - Circuit diagrams
- Electronic Components
 - Resistors
 - LEDs
- Multi-meter
- Breadboard

Introduction to Galileo (Uno +++)

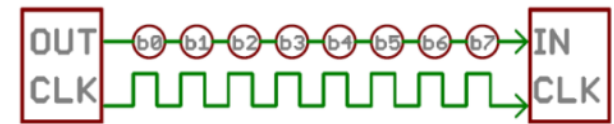
- Galileo used Arduino Rev3 as a reference and is pin compatible
- Quark SoC X1000 processor
 - x86, low-power embedded system-on-a-chip
 - 32-bit processor can run at up to 400MHz, 512 KB SRAM built-in
- 8MB Flash (to store firmware), 11KB EEPROM (non-volatile memory), μ SD socket (up to 32GB)
- Peripherals: 10/100Mb Ethernet, USB 2.0 host and device ports, RS-232 port, a mini PCI Express
- Arduino Pin out ... including six analog inputs, SPI, I2C, UART, and PWM outputs
- Software ...
 - Popular Arduino libraries like SD, Ethernet, WiFi, EEPROM, SPI, and Wire
 - You can also access the Linux side of the board with system() calls. The Linux half of the board supports stuff like Python, Node.js, SSH, Telnet, and all sorts of other, fun Linux stuff



Interact with the Linux OS

Use Remote Terminal to monitor the Galileo board and interact using standard Linux commands

- Serial Terminal Basics
 - Synchronous vs Asynchronous serial interface
 - Data bits, Synchronization bits, Parity bits, Baud rate
 - 3-wire" RS-232 connection - transmit data, receive data, and ground
- Galileo Serial Interface Cables
 - Specialized 3.5mm stereo jack to DB9 RS-232 cable ... and ... DB9 RS-232 to USB cable
 - Specialized 3.5mm stereo jack to USB cable ... and ... USB cable
- Open Port named Gadget Serial V2.4 -> UpdateDriver Software



Related Information

Datasheets Board Components:

ADC – AD7298 - <http://www.analog.com/en/analog-to-digital-converters/ad-converters/ad7298/products/product.html>

Cypress IO Expander – <http://www.cypress.com/?mpn=CY8C9540A-24PVXI>

Muxes - <http://www.ti.com/product/ts5a23159>

Software:

Arduino like IDE - https://downloadcenter.intel.com/Detail_Desc.aspx?DwnldID=23171

Putty - <http://www.putty.org/>

IoT Dev Kit (MAYBE) - <https://software.intel.com/en-us/iotdevkit>

Fritzing (MAYBE) - <http://fritzing.org/home/>

Legal Disclaimer

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: <http://www.intel.com/design/literature.htm> Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, visit Intel Performance Benchmark Limitations

All products, computer systems, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

Celeron, Intel, Intel logo, Intel Core, Intel Inside, Intel Inside logo, Intel. Leap ahead., Intel. Leap ahead. logo, Intel NetBurst, Intel SpeedStep, Intel XScale, Itanium, Pentium, Pentium Inside, VTune, Xeon, and Xeon Inside are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Intel® Active Management Technology requires the platform to have an Intel® AMT-enabled chipset, network hardware and software, as well as connection with a power source and a corporate network connection. With regard to notebooks, Intel AMT may not be available or certain capabilities may be limited over a host OS-based VPN or when connecting wirelessly, on battery power, sleeping, hibernating or powered off. For more information, see <http://www.intel.com/technology/iamt>.

64-bit computing on Intel architecture requires a computer system with a processor, chipset, BIOS, operating system, device drivers and applications enabled for Intel® 64 architecture. Performance will vary depending on your hardware and software configurations. Consult with your system vendor for more information.

No computer system can provide absolute security under all conditions. Intel® Trusted Execution Technology is a security technology under development by Intel and requires for operation a computer system with Intel® Virtualization Technology, an Intel Trusted Execution Technology-enabled processor, chipset, BIOS, Authenticated Code Modules, and an Intel or other compatible measured virtual machine monitor. In addition, Intel Trusted Execution Technology requires the system to contain a TPMv1.2 as defined by the Trusted Computing Group and specific software for some uses. See <http://www.intel.com/technology/security/> for more information.

†Hyper-Threading Technology (HT Technology) requires a computer system with an Intel® Pentium® 4 Processor supporting HT Technology and an HT Technology-enabled chipset, BIOS, and operating system. Performance will vary depending on the specific hardware and software you use. See www.intel.com/products/ht/hyperthreading_more.htm for more information including details on which processors support HT Technology.

Intel® Virtualization Technology requires a computer system with an enabled Intel® processor, BIOS, virtual machine monitor (VMM) and, for some uses, certain platform software enabled for it. Functionality, performance or other benefits will vary depending on hardware and software configurations and may require a BIOS update. Software applications may not be compatible with all operating systems. Please check with your application vendor.

* Other names and brands may be claimed as the property of others.

Other vendors are listed by Intel as a convenience to Intel's general customer base, but Intel does not make any representations or warranties whatsoever regarding quality, reliability, functionality, or compatibility of these devices. This list and/or these devices may be subject to change without notice.

Copyright © 2014, Intel Corporation. All rights reserved.

