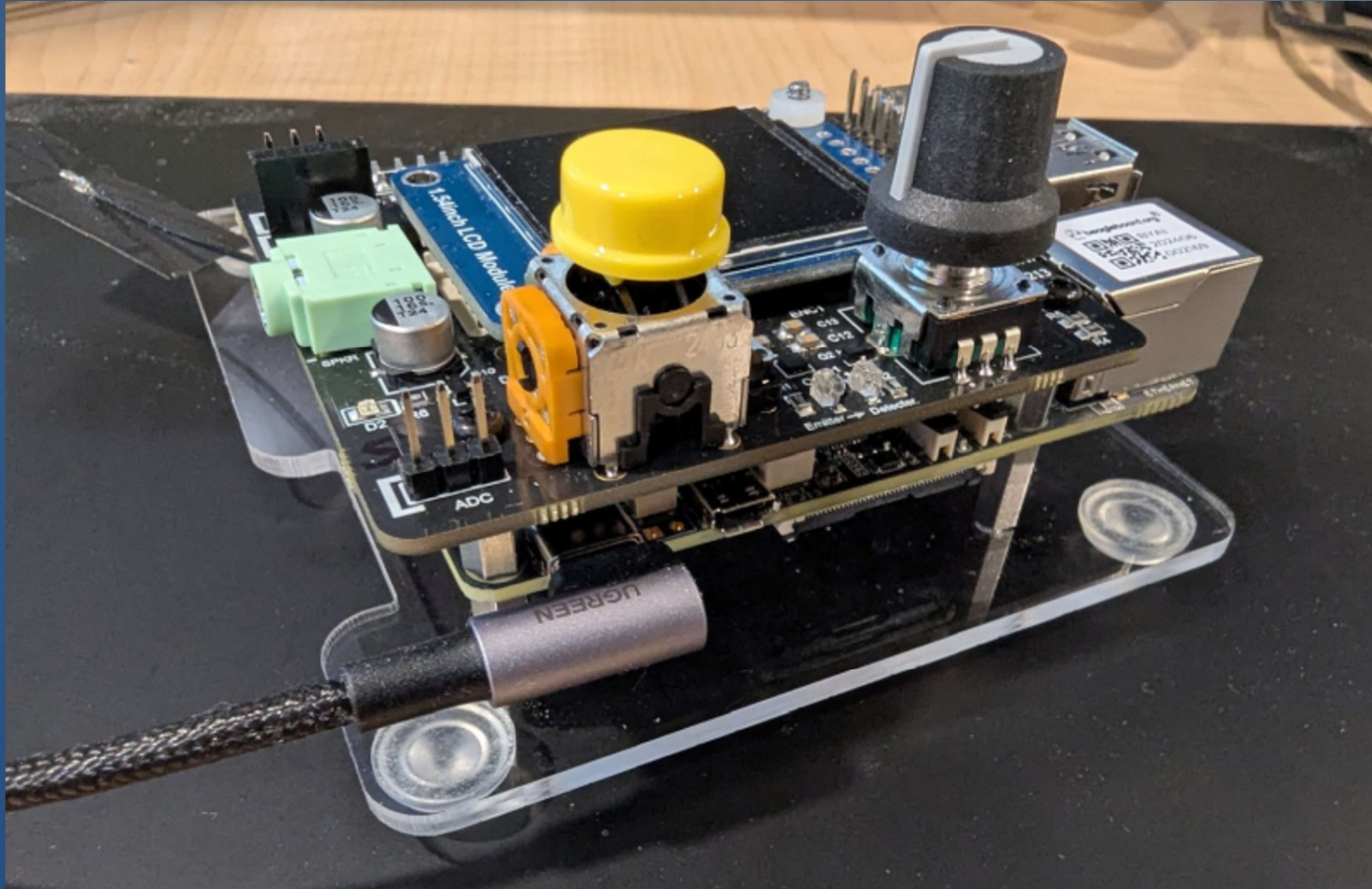


# Development Environment

## Embedded Linux Primer Ch 1&2



# Topics

- 1) Systems: Host and Target
- 2) Host setup
- 3) Host-Target communication

# Host and Target

# Host & Target

- **Host**
  - Development PC
- **Native Compiler:**
  - Run compiler on host to build for host:  
`$ gcc hello.c -o hello`
- **Cross Compiler:**
  - ..  
`$ aarch64-linux-gnu-gcc hello.c -o hello`
  - Many "cross" tools: Run on host, work with target:  
Ex: `aarch64-linux-gnu-gdb`
- **Target**
  - Embedded device

.. \$ means  
Linux prompt  
on host.



# Host & Target

- Tool naming:

aarch64-none-linux-gnu-gcc

target  
architecture

vendor  
(company)  
Optional

GNU EABI  
(embedded ABI)

gcc =  
Tool

- ABI:...

- Standard specifying how the program will:
  - layout data types in memory
  - ..  
(passing arguments, returning values).
  - perform system calls

# Host vs Target Resources

Resource	Host	Target (BeagleY-AI)
OS	Debian 13.x	Debian Linux 13.x
CPU	~3Ghz AMD64? 12-core x64	1.4Ghz ARM Cortex-A53, 4-cores, 64 bit 4 TOPS AI co-processor + realtime units
RAM	32 GB	4 GB
Storage	4,000 GB harddrive	micro SD Card
Screen	23" LCD, multi-monitor	None (1.5" LCD on Zen Hat)
Input	Keyboard, mouse	1 button, USB
Audio	In/out	Possible through USB
Ethernet	Gigabit	Gigabit
Other	DVD, Card reader	uSD Card, GPIO, I2C, SPI,...
Terminal	Screen & Keyboard	SSH & Serial
Cost	~\$1,200	~\$120

Good to match  
Linux distro on  
host to target.

# Working with Hardware

- Many embedded systems run on custom hardware.
- Interact with the world using:
  - **GPIO**:...  
Set a pin to be on (3.3V) / off (0V), or read it.
  - **I<sup>2</sup>C**:...  
Communicate with chips like an accelerometer.
  - **ADC**:...  
Read analog voltages (Ex: battery voltage).
  - **PWM**:...  
Generate a sort-of analog voltage.
- For us, the Zen Hat allows us to use all of these!

# Host Setup

# Host Setup

- Run Linux.
  - A definition of "Crazy":  
Developing for embedded Linux on non-Linux host.
- Run Linux as main OS, or in virtual machine (VM).
  - VirtualBox and VMWare Workstation / Fusion:  
lets you run Linux inside Windows/MacOS.
  - Selectively configure resources the VM gets.
  - Able to run multiple VM's on one machine.

Confusion:

Host PC: Computer you code on.

Host OS: *In VM context* means "real" OS on computer.

# Basic Linux Commands

Command	Description	Examples
ls	Directory listing. Arguments: -a for all, -l for long (all info)	ls ls -l
pwd	Show current directory name	pwd
mkdir	Make a directory	mkdir myNewPlace
cd	Change directory	cd myDir                      cd \myDir cd ..                          cd \
chmod	Change file permissions	chmod a+r hello.a
chown	Change file owner	chown bfraser hello.a
sudo	Execute as administrator	sudo chown bfraser hello.a
apt-get	Install a program	sudo apt-get install somepackage
gedit	Edit a file (new window)	gedit hello.cpp &
ifconfig	Configure networking	ifconfig eth0 192.168.0.1
mount	Mount a file-system	mount -t nfs \ 192.168.0.103:/opt/img /mnt/img
nano	Edit a file in the terminal	nano hello.cpp



# Basic Linux Commands

Command	Description	Examples
cat	Dump to screen	<code>cat hello.cpp</code>
less	Show on screen with "more.." prompt. ('q' to quit)	<code>less hello.cpp</code>
tar	Archive management (unzip)	<code>tar xvfj hello.tar.jz2</code>
find	List all files in sub-folders	<code>find</code>
grep	Search for a string	<code>grep "Hello world" *.cpp</code>
 (Shift \)	Pipe: redirect output to second program's input	<code>find   grep hello.cpp</code>
>	Redirect output to a file	<code>ls &gt; listing.txt</code>
rm	Remove a file (delete)	<code>rm listing.txt</code>
echo	Print some text	<code>echo hello</code>
dmesg	Show kernel boot messages	<code>dmesg</code>

Recommended Linux Shell Tutorial:

Ryan's Tutorials: <https://ryanstutorials.net/linuxtutorial/>

# Communication

How can we access the target?  
**We need a Linux terminal, but how?**



Micro-B plug, xxx, Mini-B plug, Standard-A receptacle, Standard-A plug, Standard-B plug

# Serial & Ethernet

- We can connect to the BYAI via a **Serial Port** and **Ethernet**
- **Serial Port**
  - A very low level communication port (need external hardware)
  - Used for sending characters between target and PC
  - Slow, but..
- **Ethernet (over USB connection!)**
  - Fast network connection
  - Used for **SSH** (Secure Shell) for a terminal
  - Used for **NFS** (Network File System) to share files
  - ..

# Communications Overview

- **Serial Port:**

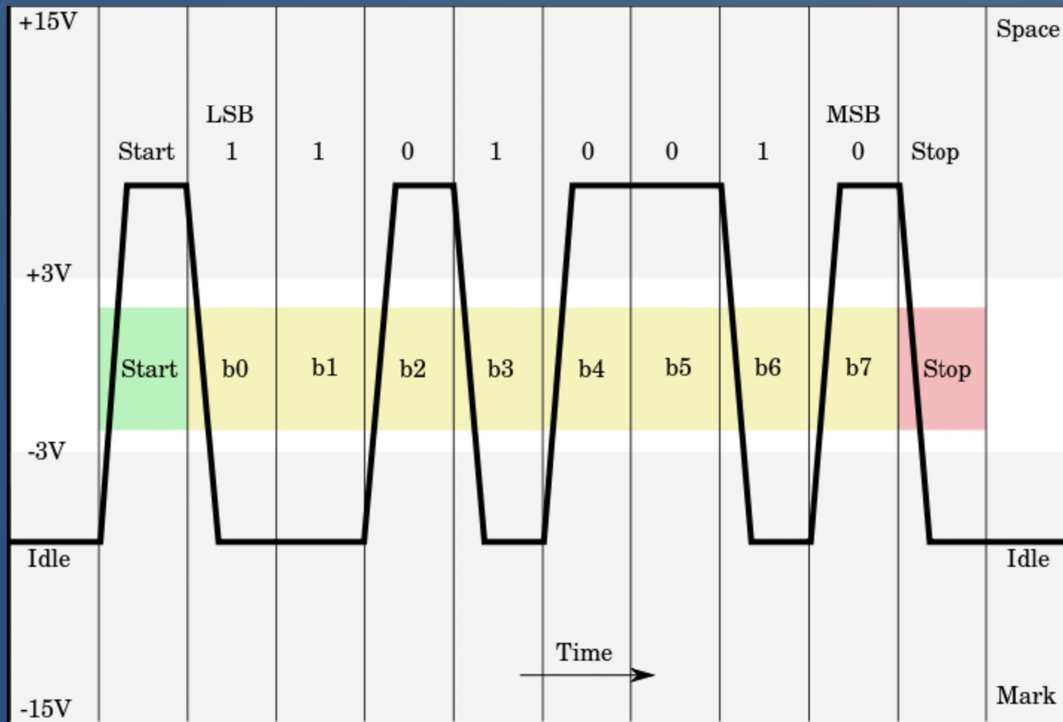
- Access target's Linux terminal via the serial port  
(Need serial port when can't use SSH: booting or errors)
  - Serial protocol for +/-12V
  - 0-3V (or 0-5V) serial protocol.
  - BYAI has port for a Raspberry Pi Debug Probe
- Host uses “**Screen**” program show serial data.

- **Ethernet Network:**

–



# RS232 Serial Protocol

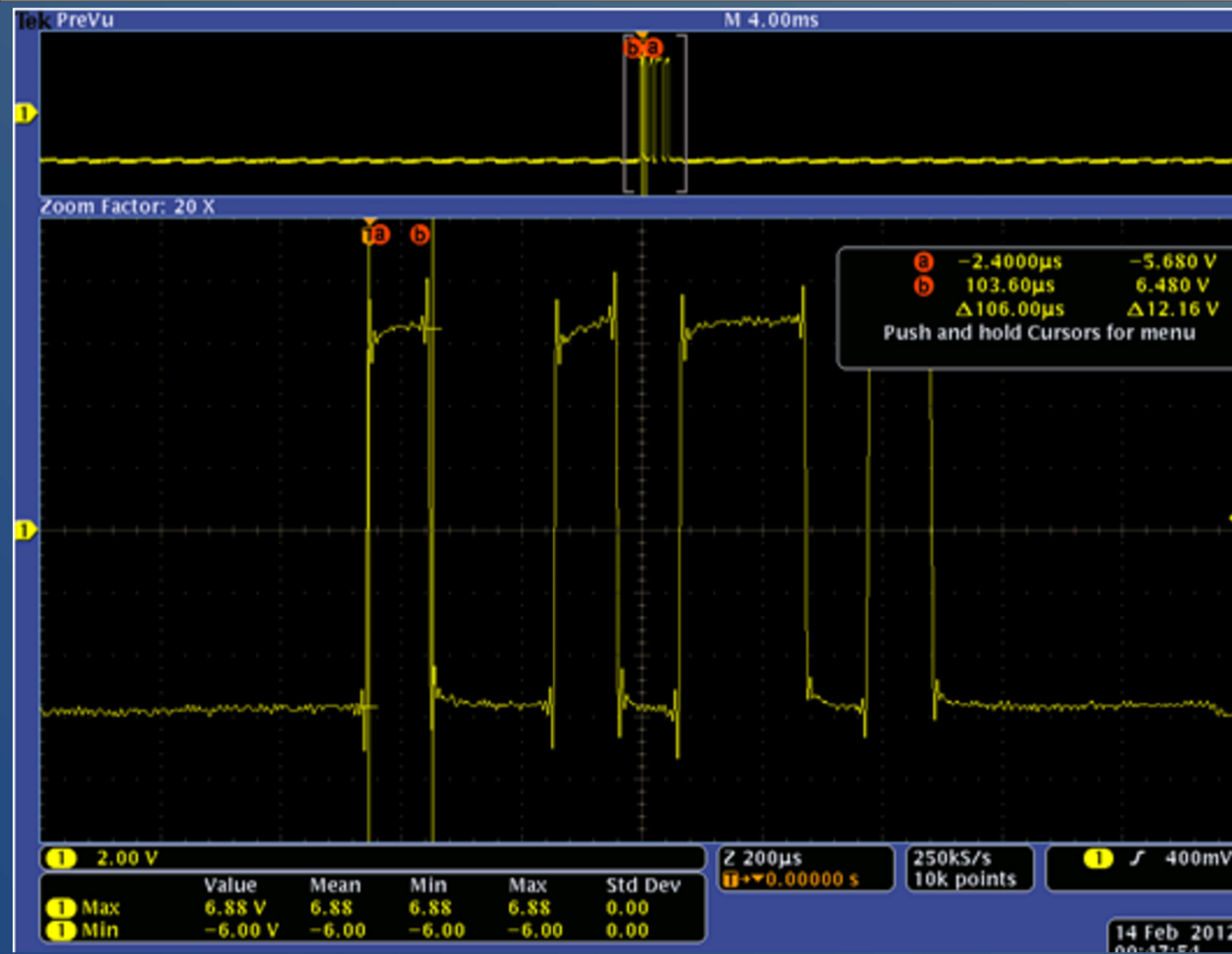


- **RS232 voltages**
  - $-12V = 1$ ;  $+12 = 0$
- Each bit has fixed time dependant on **bitrate**
- Starts with **start bit(s)** (+12v); ends with **stop bit(s)** (-12v)
- Diagram shows a 'K' character (0x4B)

RS = "Recommended Standard"



# RS232 Real World View



- Oscilloscope trace of 'K' (0x4B), 1 start bit, 8 bits, 2 stop bit.
- HW has to be sync'd to know where to sample each bit.
- Timing errors lead to garbage characters.



# RS-232 & TTL Settings

- **RS-232 is a "Serial Port"**
  - Connector is often a **DB9 / DB25** with **bi-directional data**: can Tx and Rx at the same time.
- Zen cape has FTDI chip to convert TTL serial data from the microprocessor to USB; host has drivers to access it like a "normal" serial port. (Called **TTL-232**)
- **BeagleY-AI Serial Port Settings**
  - Speed (bps), #bits/byte, parity check, # stop bits:  
.. **< 14kBytes / second!**
  - Optional **handshaking** to control data transmission.
    - We'll always use no handshaking



# Screen Program

- Run *Screen* on host to view target's serial port.

- Screen Usage on host

- Install: `$ sudo apt-get install screen`
  - Run: `$ sudo screen /dev/ttyACM0 115200`

Don't type \$

No sudo access in lab host OS.  
(Can run screen there without sudo)

- Screen Operations

- Show help: Control a + (no control) ?
  - Quit: Control a + (no control) \

- Linux Ports

- `/dev/ttyACM0` how Linux supports TTL over USB

To show kernel messages when  
USB device connected:..

# Network

- BeagleY-AI can network in 3 ways:
  - Ethernet over USB
    - USB-C cable allows BeagleY-AI to mount on host PC as a network connection.
    - Host has IP: 192.168.7.1
    - Target has IP: 192.168.7.2
  - WiFi
    - Built-in WiFi support
  - Ethernet
    - “RJ45” Ethernet connection.
    - BYAI uses DHCP to get an IP address:  
DHCP =..

# Networking Basics

- Find out IP settings:

```
(host) $ ip addr
```

```
(byai) $ ip addr
```

(host): means host PC command  
(byai): means target command

- ssh to open a terminal to the target

```
(host) $ ssh debian@192.168.7.2
```

- ping to test TCP/IP connection to board:

```
(host) $ ping 192.168.7.2
```

```
(byai) $ ping 192.168.7.1
```

# Files over the Network

- Mounting directory over NFS
  - NFS:...
  - Use NFS to make application testing MUCH faster:
    - Transferring ~50 meg takes ~1min vs ~1hr.
    - On the target, mount the host's directory and..
- “Pro” Tip:  
Always look for ways to make development faster.

# Review

1. What is cross compiling?
2. What does **sudo** do?
3. What can we use a **serial connection** for?
4. Explain why **NFS** is useful.



# Summary

- Develop on **host**, deploy on **target**.
  - **Cross compile** on host for target.
- Target has **limited resources**, but custom hardware:
  - **GPIO, I2C, A2D, PWM.**
- Host running **Linux in VM or native**
- Communicate to target using **Ethernet**
  - **ssh, Ping, NFS.**

# Flash Operation

- Flash has pages and blocks
  - Page ~2K bytes, Block ~128K bytes (=64 pages)
  - - Page erase sets it to all 1's.
    - Writing can leave the 1, or set to 0.
    - Cannot "write"...  
must erase page.
- Lifetime
  - Pages have...
    - Almost always have some bad blocks.
  - Must use special file-systems to do wear-levelling...  
(not FAT or NTFS)