>warpx.io

> Rapid IoT & Wearable Prototyping featuring Hybrid Architecture

Nicola La Gloria
Aaron Moore
warpx.io

Wearable Tech Conference, San Jose 2016

Overview

- Talk today focuses on the concepts learned by our team in developing prototypes in both IoT and wearables

- We will be presenting a solution that we developed with the support of leading companies in this industry

- Concepts used here can be applied to many platforms

Why do we need to build small devices

- Start-ups/Companies/Makers/Researchers in the IoT and wearables are needing to build increasingly smaller electronics

- Can be required to prove concepts before getting approval/funding/etc, but how?

- To design this, they face challenges traditionally left to large companies with big budgets

Challenges of building small

- Building form-factor devices introduces many new layers of complexity into the design. Not just electronics but also the enclosure and back-end connectivity, support peripherals, UX design, etc.

- Typical traditional design cycles for these systems are long

- Large customers have custom parts made just for them to help scale their designs. Resources not always available to broad audience

Challenges of building small

- Need to give developers the right tools, same or similar to those used by big companies

- Existing tools that are fast, do not scale, or scale, but do not fit well for IoT & wearables due to missed considerations (driven by different market needs).

- We realized the need for a process that allows makers and professionals to #buildfast proof of concepts, reducing dev time and cost.

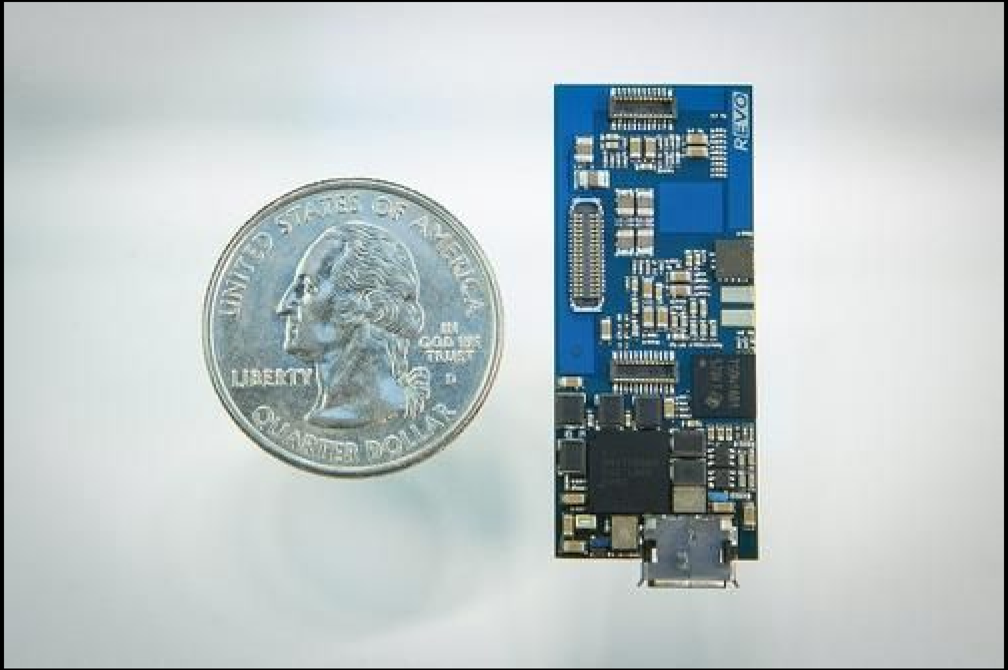
MCU vs AP

- Designing with Microcontrollers vs an Application Processor is very different.

- Both are good choices and depends on the design of the Wearables or IoT device.

- However, APs can save valuable development time early in the design stage by providing a lot of functionality that is typically difficult to implement well on an MCU (reduces design time for prototypes).

What we developed...



warpx

- The idea of warpx is the design of small form-factor, powerful embedded systems designed around an architecture we developed called Hybrid Design.
- warp_0x01 is the first of this family of products designed and built by us.
- Our hope/goal is to help pave way for the development of small and powerful computing devices integrated into wearables, sensors, and various other IoT devices.

A Rapid prototyping toolbox

- warp_0x01 is the core upon which designs can be built and provides the part that is most difficult in terms of development (both design and manufacturing).

- 80% of the needed functionality comes out of the box. Just add your sensor.

- warp_0x01 is open: Software, Hardware, BOM, schematics. You can manufacture it too.

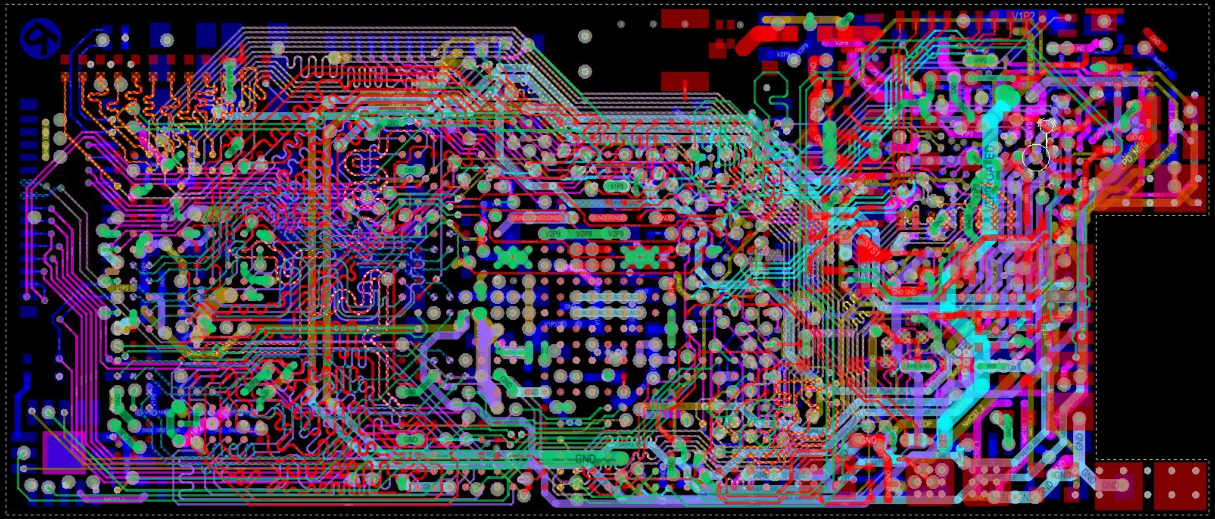
warpx.io \O/

- warpx.io is the community hub embracing today all the developers that actively developed the codebase.

- Aims to be the home for makers, professionals, designers, developer, blogger working with these small devices and whoever wants to contribute to this project.

Main contributors: Diego Rondini, Ray Anderson, Jacob Postman, Otavio Salvador, Eric Nelson, David Clack, Elena Contini, Will Martindale, Nicola La Gloria, Aaron Moore

...& you :)

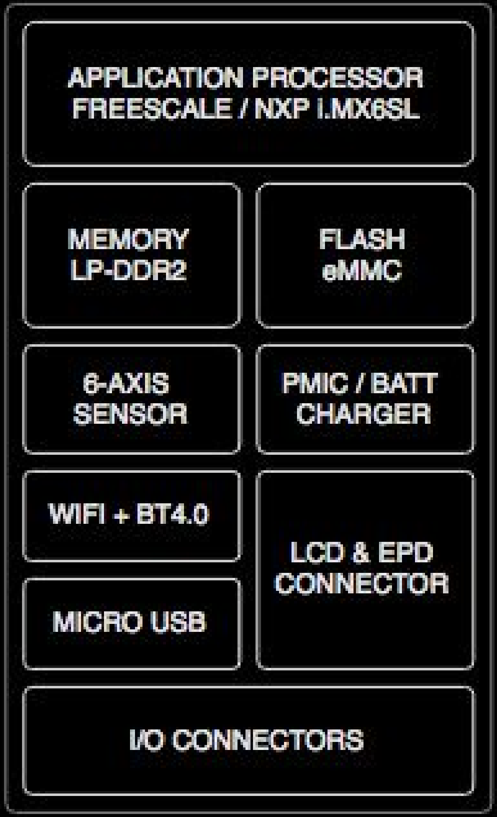


16mm

38mm

Hardware Details:

- Small form factor: 16mm x 38mm x 3.5mm (Board Area < 1 sq-in)
- 10 Layer HDI PCB (0.4mm BGA, LGA, 0201)
- 1Ghz ARM Cortex-A9 (Freescale i.MX6SL)
- 512MB Memory + 4GB Flash
- USB OTG, 6-axis ACC+Mag, PMIC, Wifi+BT/BLE
- B2B Expansion



APPLICATION PROCESSOR
FREESCALE / NXP I.MX6SL

MEMORY
LP-DDR2

FLASH
eMMC

6-AXIS
SENSOR

PMIC / BATT
CHARGER

WIFI + BT4.0

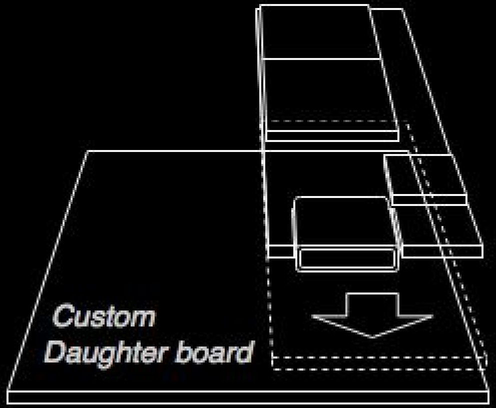
MICRO USB

LCD & EPD
CONNECTOR

I/O CONNECTORS

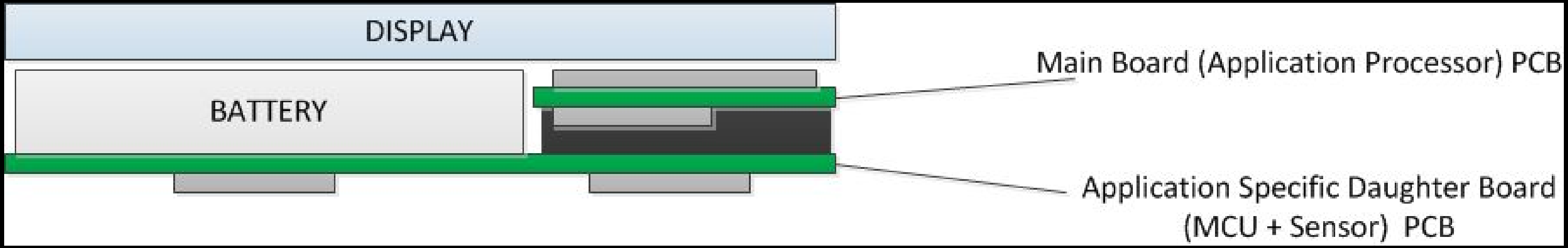
Hybrid Architecture

- Prototyping hardware with APs is hard, MCUs are much faster.
- Hybrid Architecture responds to the need of having a design featuring the power of an AP and the low power capability of a MCU



Hybrid Architecture

- Mainboard (AP): reusable high performance core
- Daughter Board (MCU): application specific payload (typically just a sensor)


DISPLAY
BATTERY
Main Board (Application Processor) PCB
Application Specific Daughter Board (MCU + Sensor) PCB

The Puzzle Game

Hardware:

- Warp
- Daughter Board (can be custom)

Development Tools:

- Interposer Board

Accessories:

- Displays
- Battery
- Sensors (many)

OS:

- Yocto Linux
- Android

Software:

- Java, Python

I/O & Communication:

- USB OTG
- BT, BLE
- WIFI 802.11 b/g/n
- Ethernet & serial gadget

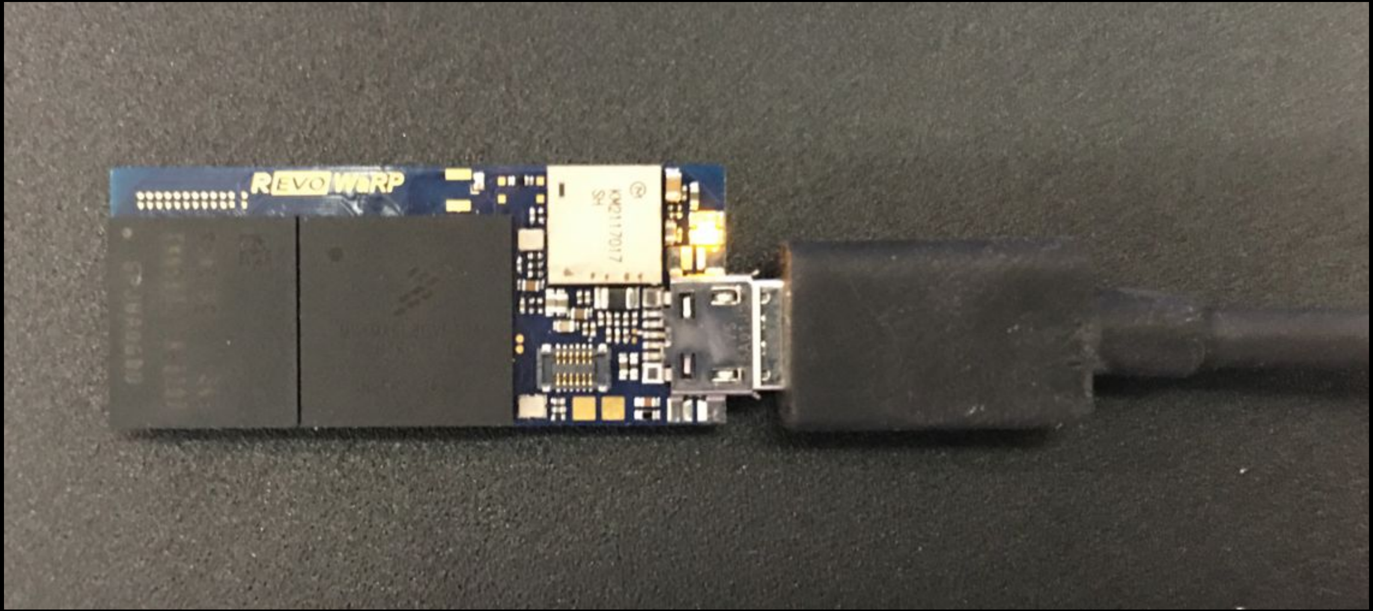
Prototyping Scenarios

- Using the puzzle pieces we assemble a variety of configurations that build up the prototype.

- The steps and setups will help to determine your design needs and can be useful to prove the validity of a concept very early.

- Using these methods, the focus is always on your application specific design needs, never on the design or re-development of tools.

Get Started



Get Started

- Hardware: Warp
- Software: Yocto Linux
- I/O: WiFi, Serial Gadget

- At this stage, very similar to every other SBC (Rpi, Wandboard etc,) except that warp is µ, very µ

Get Started

- Basic computing platform, perhaps as a gateway or hub.
- Linux tools allow network code to be written easily
- Can start to grab data (aggregate) from local devices (BT/WiFi) and processed this on-board (storage, algorithms, connection to cloud).

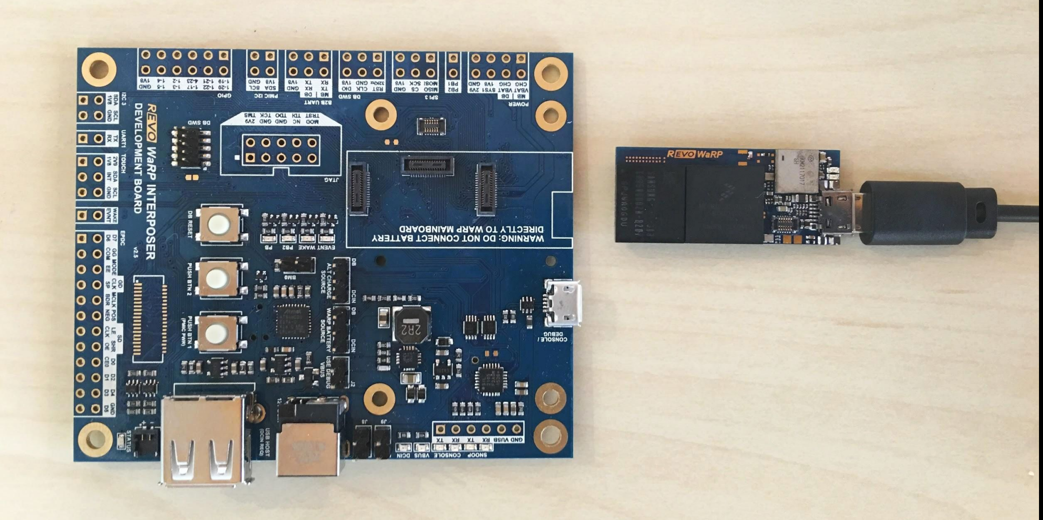
Hacking Around

- Hardware: Warp, Arduino
- Dev Tools: Interposer Board
- OS: Yocto Linux
- I/O: WiFi, Serial Gadget
- Software: Python

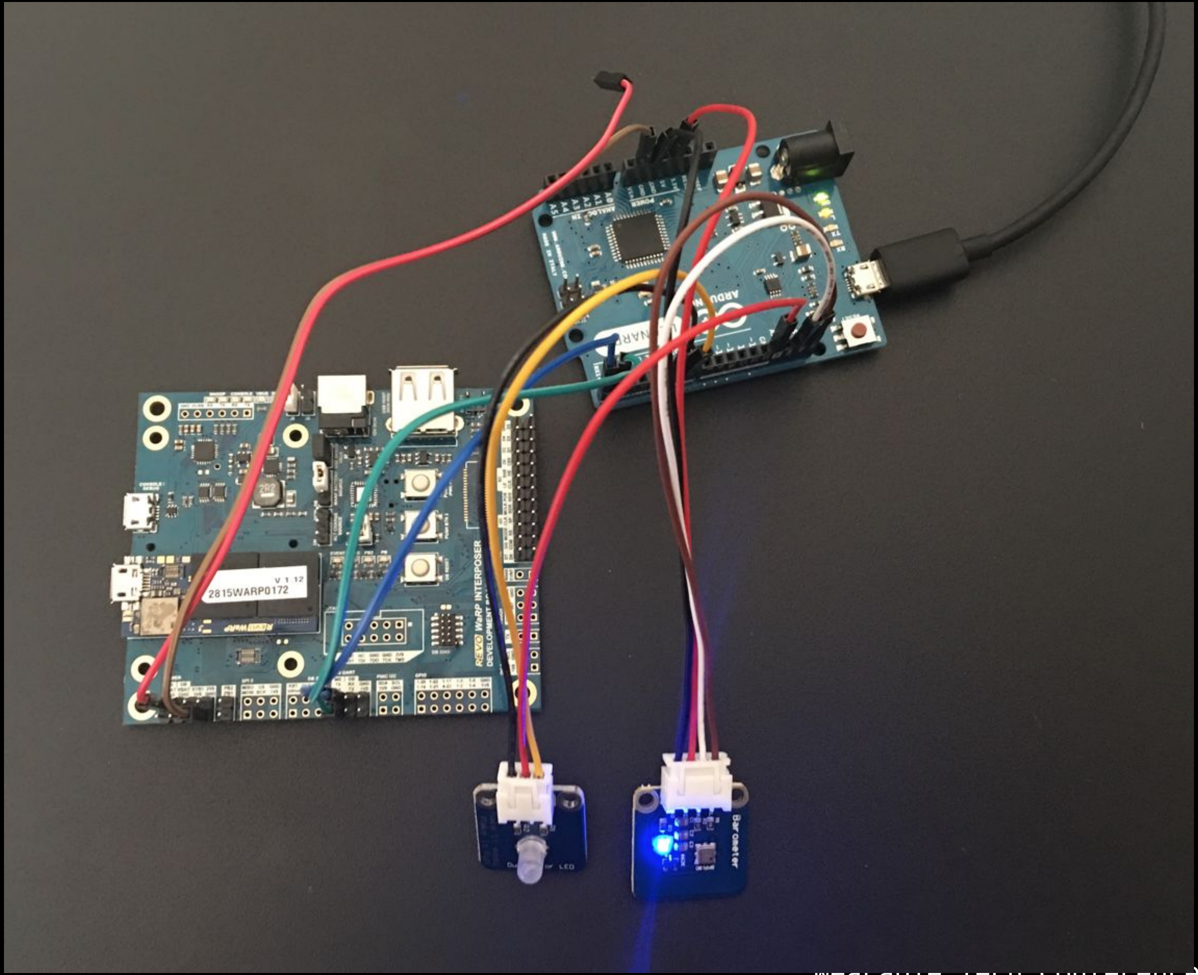
Arduino - off the shelf, simple to hook to warp without reinventing everything. Hands on with sensors and controls

Python - Real easy access to peripheral boards.

Hacking Around



- the Interposer makes development faster & easier


2815WARP0172
V 1.12
WaRP INTERPOSER
DEVELOPMENT BOARD
REVO
Barometer
ARDUINO

Hacking Around

- Arduino or similar HW familiar to many developers. Keeps MCU code VERY simple for accessing sensor. Lots of existing code.

- Many sensors to play with, but might not be the exact sensor you want to use. Benefit is no custom hardware is needed.

- Few lines of code (both MCU and AP) gets data streaming from SENSOR -> MCU -> AP -> UI or Cloud

- Problem is that it’s too big to do anything small. That’s OK.

Prototype 0x01 (headless)

- **Hardware:** Warp, custom sensor board (hooked to the interposer pin outs)
- **Dev Tools:** Interposer Board
- **OS:** Yocto Linux
- **I/O:** WiFi, Serial Gadget
- **Software:** Python, Java, APIs

Value: Early stage IoT headless edge device proof of concept && get application development started (issue is often the disconnection between hardware and software teams)

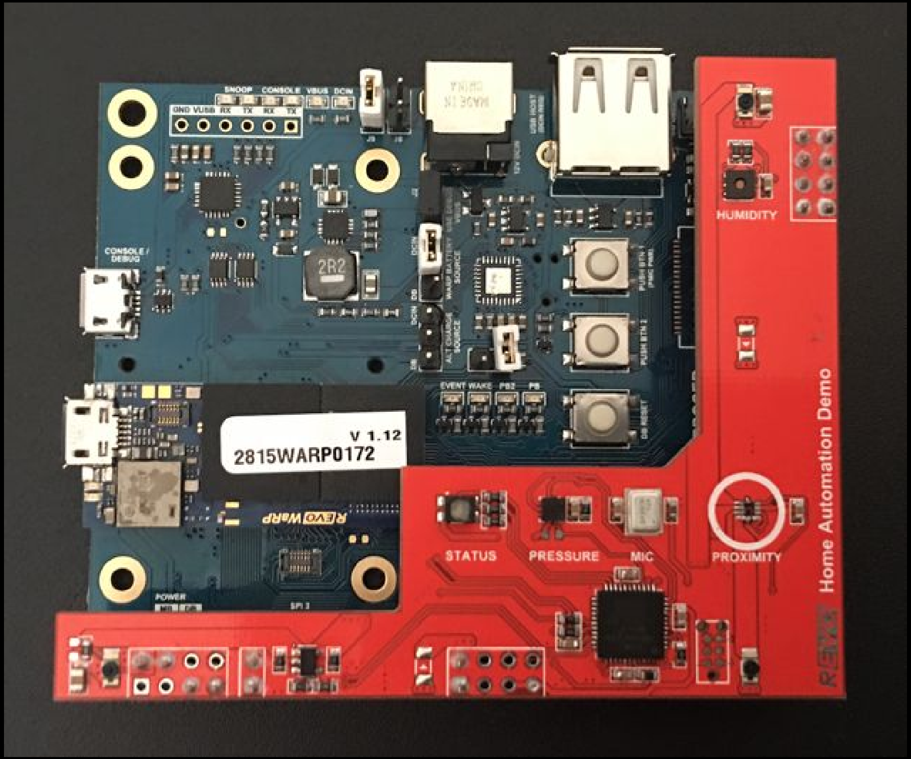
Prototype 0x01: defining the API

- Define APIs
- API is the contract between hardware and software

Value: hardware can continue to iterate and software development is not at the end of the cycle (as always :Q)

Example (JSON payloads):

```
    { “RGB_LED” : [R,G,B] }           //0-255 values for rgb

    { “TEMPERATURE” : FLOAT_C }
```


HUMIDITY
STATUS
PRESSURE
MIC
PROXIMITY
Home Automation Demo
CONSOLE / DEBUG
V 1.12
2815WARP0172
REVO WaRP

Prototype 0x01 (headless)

- Gets hardware functional with the application specific sensors onto a rough board that can be replicated easily. No wires

- Easy and fast to develop hardware like this. Very low cost from hardware perspective.

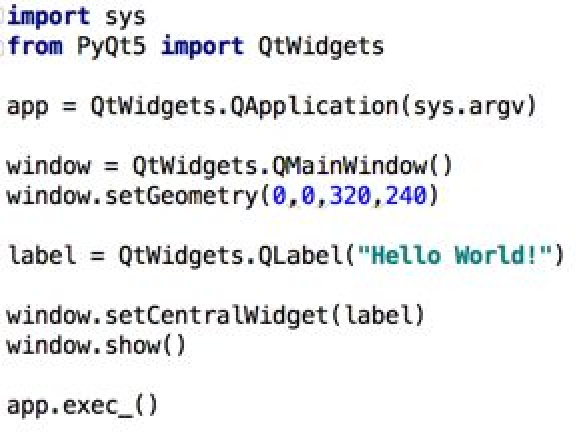
- Plugs into existing dev tools so application team can get access to hardware early.

Prototype 0x02 (headful)

- Hardware: Warp, custom sensor board (hooked to the interposer pin outs)
- Accessories: Display
- Dev Tools: Interposer Board,
- OS: Yocto Linux
- I/O: WiFi, Serial Gadget
- Software: Python, Java, APIs

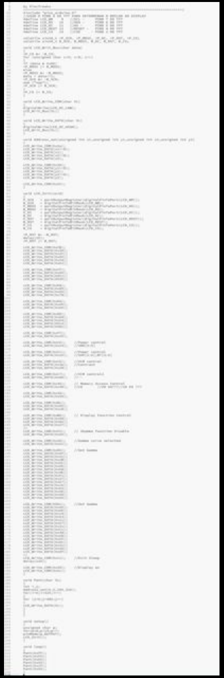
Value: Early stage IoT #HEADFULL edge device proof of concept.

“Hello World” GUI code



```python
import sys
from PyQt5 import QtWidgets

app = QtWidgets.QApplication(sys.argv)

window = QtWidgets.QMainWindow()
window.setGeometry(0,0,320,240)

label = QtWidgets.QLabel("Hello World!")

window.setCentralWidget(label)
window.show()

app.exec_()
```

With pyQT on AP

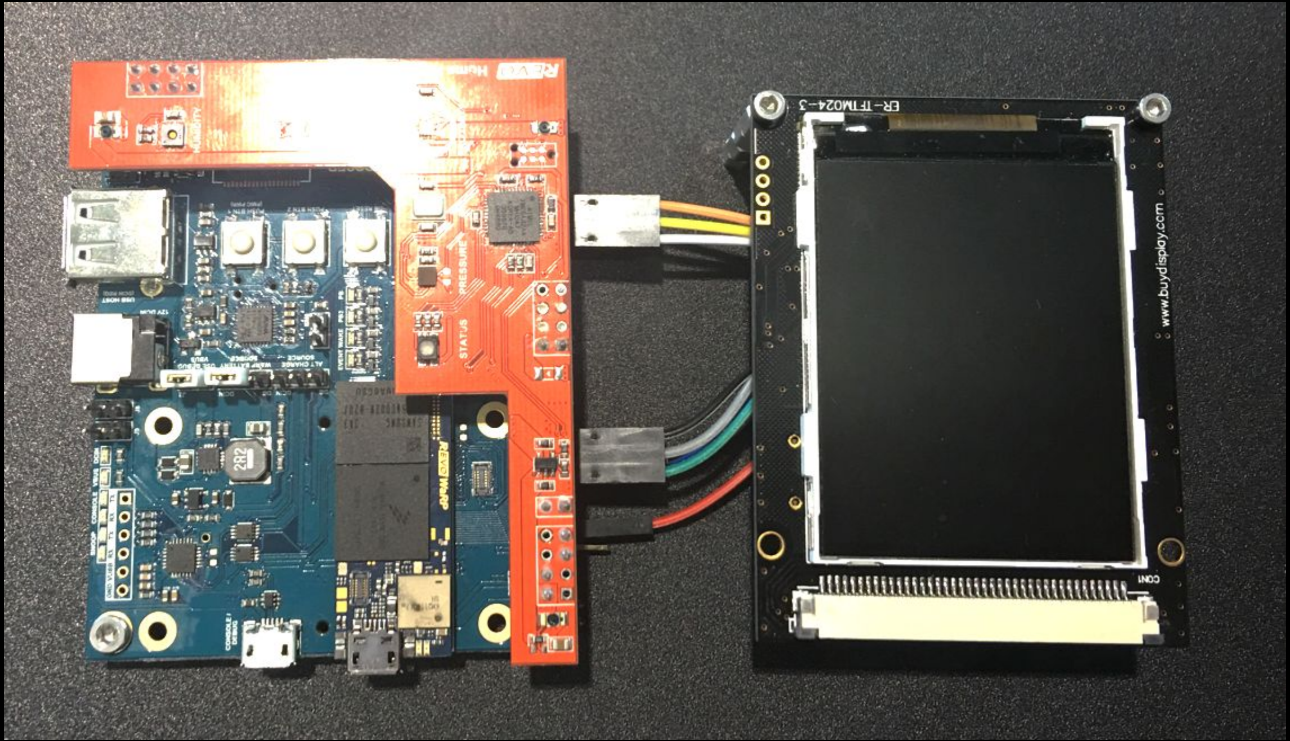


MCU code on same display

Prototype 0x03

- Hardware: Warp, ~~custom sensor board (hooked to the interposer pin outs)~~, small form factor custom daughter board
- Accessories: Display, Battery
- Dev Tools: ~~Interposer Board~~
- OS: Yocto Linux
- I/O: WiFi, Serial Gadget
- Software: Python, Java, APIs

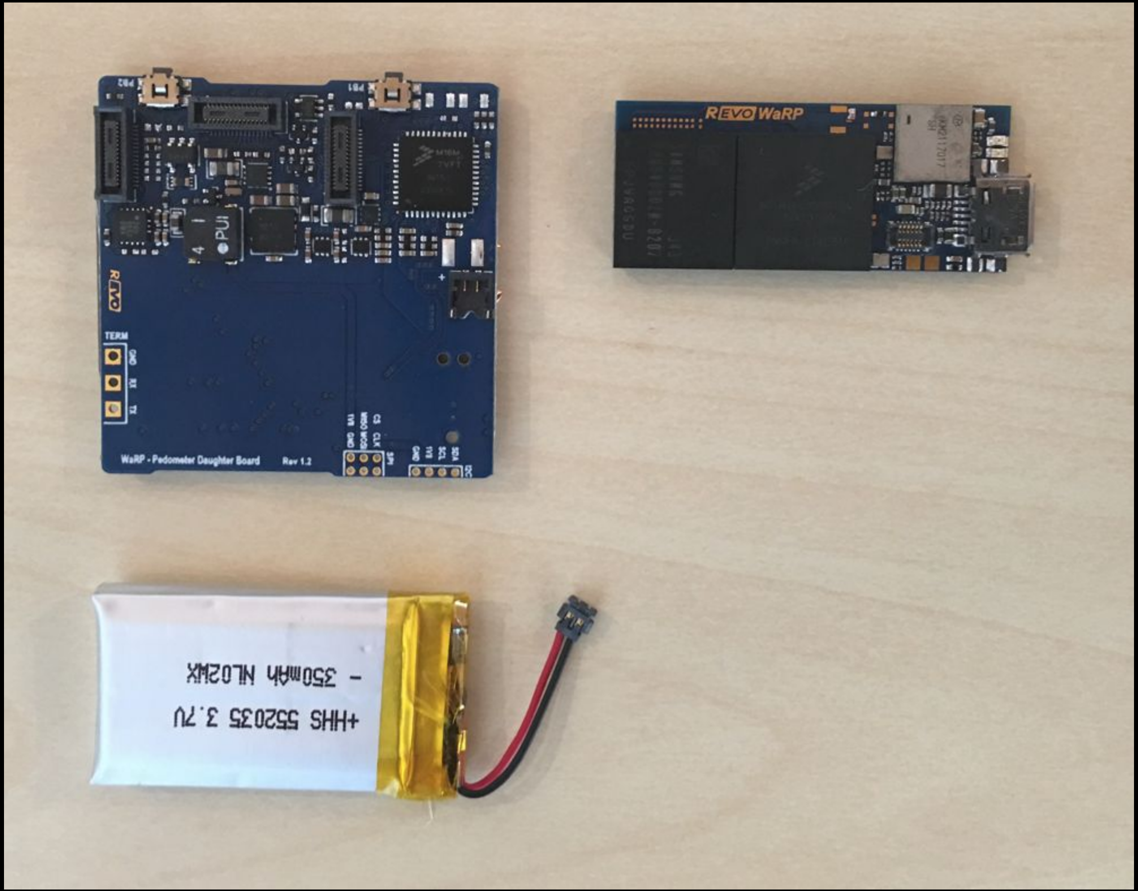
Value: proof of concept of an optimized IoT headful edge device


ER-TFTM024-3
www.buydisplay.com
CON1
PRESSURE
STATUS
2R2
WaRP
CONSOLE / DEBUG
USB HOST

Prototype 0x04

- ❯ Hardware: Warp, #pedometer daughter board concept
- ❯ Accessories: Display, Battery
- ❯ Dev Tools: ~~Interposer Board~~
- ❯ OS: ~~Yocto Linux~~, Android
- ❯ I/O: WiFi, Serial Gadget, BT
- ❯ Software: Java

Value: proof of concept of a #headful wearable device


REVO WaRP
TERM
GND
RX
TX
WaRP - Pedometer Daughter Board
Rev 1.2
+HHS 552035 3.7V
- 350mAh NL02WX

REVO

Productize it

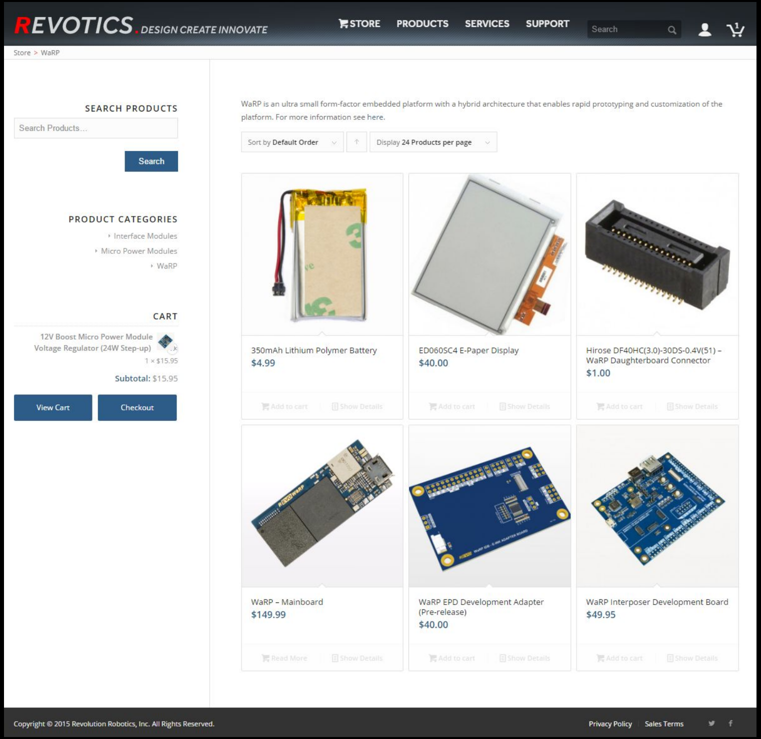
- In the workflow (from Getting Started to 0x03) you see definitively steps common to take concept to productization.

Such as:

- IoT Sensor (like Nest)
- Home Automation (like Amazon Echo)
- eInk Reader/Signage (Kindle)
- Medical Devices

Availability

- ❯ Direct from us
- ❯ Shipping today
- ❯ More coming soon



WaRP is an ultra small form-factor embedded platform with a hybrid architecture that enables rapid prototyping and customization of the platform. For more information see here.

12V Boost Micro Power Module Voltage Regulator (24W Step-up)
1 × $15.95

Subtotal: $15.95

350mAh Lithium Polymer Battery
$4.99

ED060SC4 E-Paper Display
$40.00

Hirose DF40HC(3.0)-30DS-0.4V(51) – WaRP Daughterboard Connector
$1.00

WaRP – Mainboard
$149.99

WaRP EPD Development Adapter (Pre-release)
$40.00

WaRP Interposer Development Board
$49.95

License

- Hardware schematics and all documentation are licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
- Operating systems are released under their respective licenses
- No proprietary software on public codebase

References

- Mailing list: >warpx.io on Google group
- Store: http://revotics.com/store
- Community website:www.warp.io
- Documentation:www.warpx.io/resources
- This presentation: www.warpx.io/resources
- GitHub: https://github.com/warpboard

Subscribe!

>THANK YOU

> want a t-shirt?

>front

>back