UNIVERSITY OF Hull

Making Gadgets

Rob Miles

Department of Computer Science

University of Hull

Agenda

- What makes a gadget?
  - What do gadgets do that make things interesting?
- Computers for Gadgets
- Interfacing with the outside world
- Printing boxes for your gadgets
- Making your own gadgets

Gadgets all around

- The human race has always been all about tools
- From the plough, to the wheel, to the Apple iPad, we have surrounded ourselves with gadgets

What makes a Gadget?

Gadget: *Noun - A small mechanical device or tool, esp. an ingenious or novel one.*

- The first gadgets were mechanical
- You can still get mechanical ones now
- But today most gadgets have an electronic component

.NET for Multiple Device types

- You might think of C#, Visual Studio and .NET as tools that you use to create PC applications
  - Compile and run your programs and run them on your desktop or laptop
- However, there is more to .NET than this
  - You are probably aware that you can write .NET applications for your Windows Mobile device
- But now you can also target all kinds of devices

The Windows PC

Tablet PC

Notebook PC

Embedded Windows

UNIVERSITY OF Hull

Tablet PC

Notebook PC

Thin Clients

Industrial Automation

Machine Control

ATMS & Kiosks

Point of Sale

Medical Devices

Windows Embedded CE

UNIVERSITY OF Hull

Tablet PC

Notebook PC

Thin Clients

Industrial Automation

Machine Control

ATMS & Kiosks

Point of Sale

Medical Devices

Windows Mobile Smartphone

Windows Mobile Pocket PC Phone

Zune

Windows Automotive

Mobile Handhelds

VoIP Phones

Set-top Boxes

.NET Micro Framework

UNIVERSITY OF Hull

Tablet PC

Notebook PC

Windows

Thin Clients

Industrial Automation

Machine Control

ATMS & Kiosks

Point of Sale

Medical Devices

Windows Embedded Standard

Windows Mobile Smartphone

Zune

Mobile Handhelds

Windows Mobile Pocket PC Phone

Windows Automotive

VoIP Phones

Set-top Boxes

Windows Embedded CE

Sensor Networks

Auxiliary Displays

Remote Controls

Wearable Devices

Microsoft .NET Micro Framework

Platforms and Hardware

	Windows Vista	Windows XP Embedded	Windows CE	.NET Micro Framework
Device Features	Connected, PC Class Performance, Accelerated Graphics	Connected, PC Class Performance	Connected, Graphical UI, Server, Browser, RAS DirectX	Connected, Small, Wearable, Graphical UI
Development	Visual Studio + C# .NET Framework	Visual Studio + C# .NET Framework	Visual Studio + C# .NET Compact Framework	Visual Studio + C# .NET Micro Framework
Footprint	X86 Dual Core, 64 Bit, Multi Processor	X86	X86, MIPS, SH4, ARM, with MMU, 12 MB, Managed Code	ARM 7, ARM 9, No MMU, 250-500KB, Managed Code
Power	Mains	Mains	Low Power	Very Low Power

Platforms and Hardware

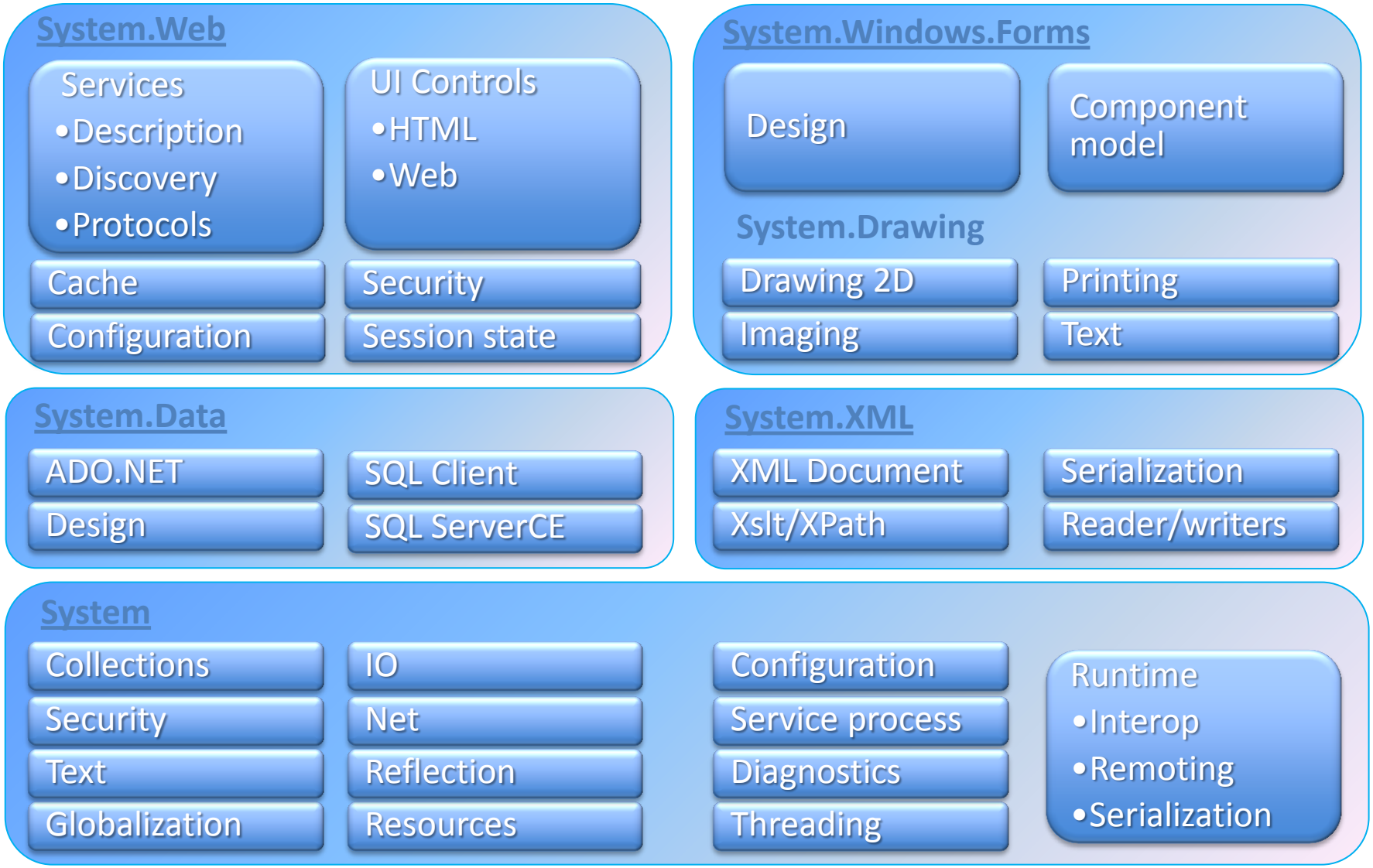
	Windows Vista	Windows XP Embedded	Windows CE	.NET Micro Framework
Device Features	Connected, PC Class Performance, Accelerated Graphics	Connected, PC Class Performance	Connected, Graphical UI, Server, Browser, RAS DirectX	Connected, Small, Wearable, Graphical UI
Development	Visual Studio + C# .NET Framework	Visual Studio + C# .NET Framework	Visual Studio + C# .NET Compact Framework	Visual Studio + C# .NET Micro Framework
Footprint	X86 Dual Core, 64 Bit, Multi Processor	X86	X86, MIPS, SH4, ARM, with MMU, 12 MB, Managed Code	ARM 7, ARM 9, No MMU, 250-500KB, Managed Code
Power	Mains	Mains	Low Power	Very Low Power

Platforms and Hardware

	Windows Vista	Windows XP Embedded	Windows CE	.NET Micro Framework
Device Features	Connected, PC Class Performance, Accelerated Graphics	Connected, PC Class Performance	Connected, Graphical UI, Server, Browser, RAS DirectX	Connected, Small, Wearable, Graphical UI
Development	Visual Studio + C# .NET Framework	Visual Studio + C# .NET Framework	Visual Studio + C# .NET Compact Framework	Visual Studio + C# .NET Micro Framework
Footprint	X86 Dual Core, 64 Bit, Multi Processor	X86	X86, MIPS, SH4, ARM, with MMU, 12 MB, Managed Code	ARM 7, ARM 9, No MMU, 250-500KB, Managed Code
Power	Mains	Mains	Low Power	Very Low Power

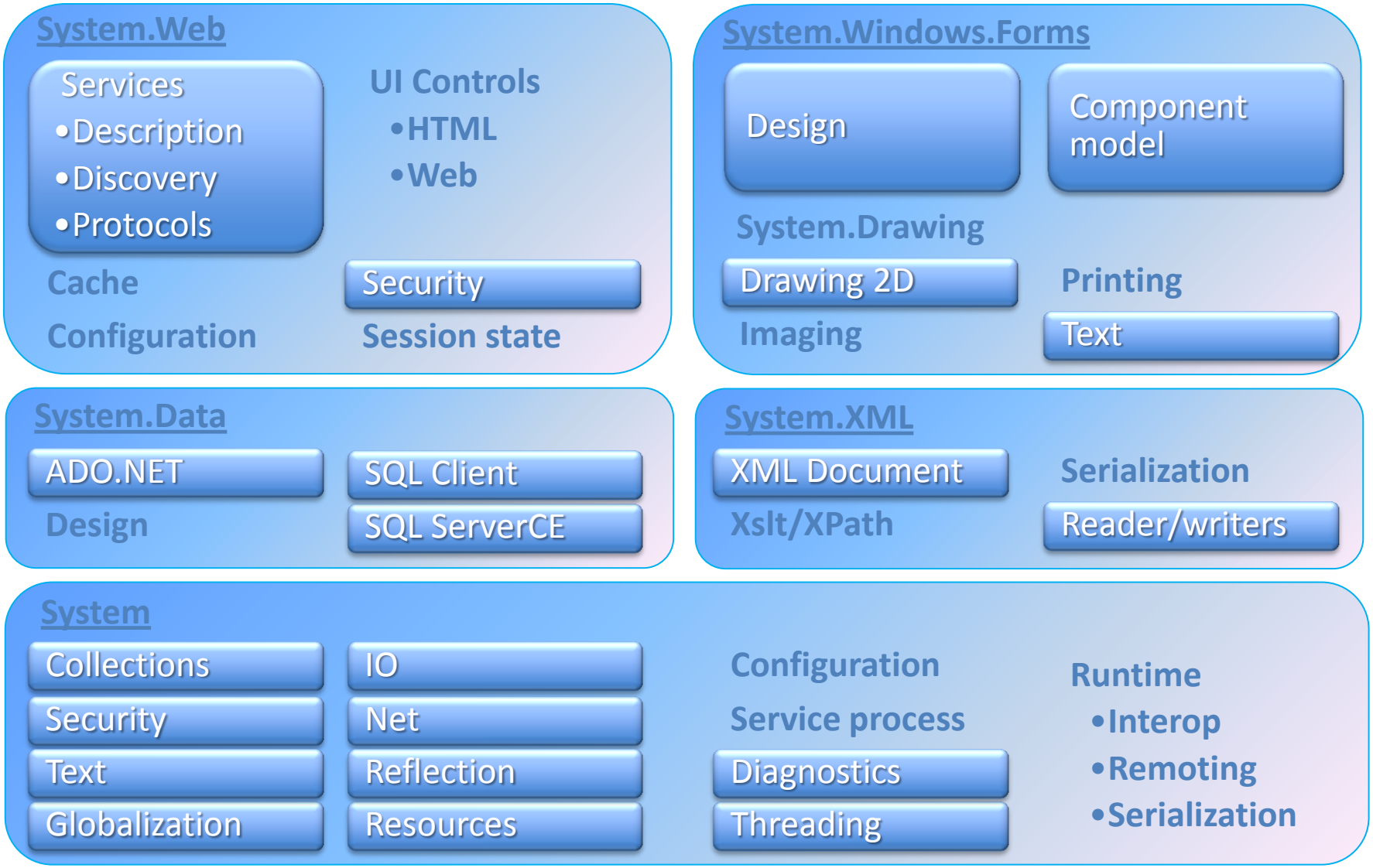
.NET Framework on Windows

University of Hull



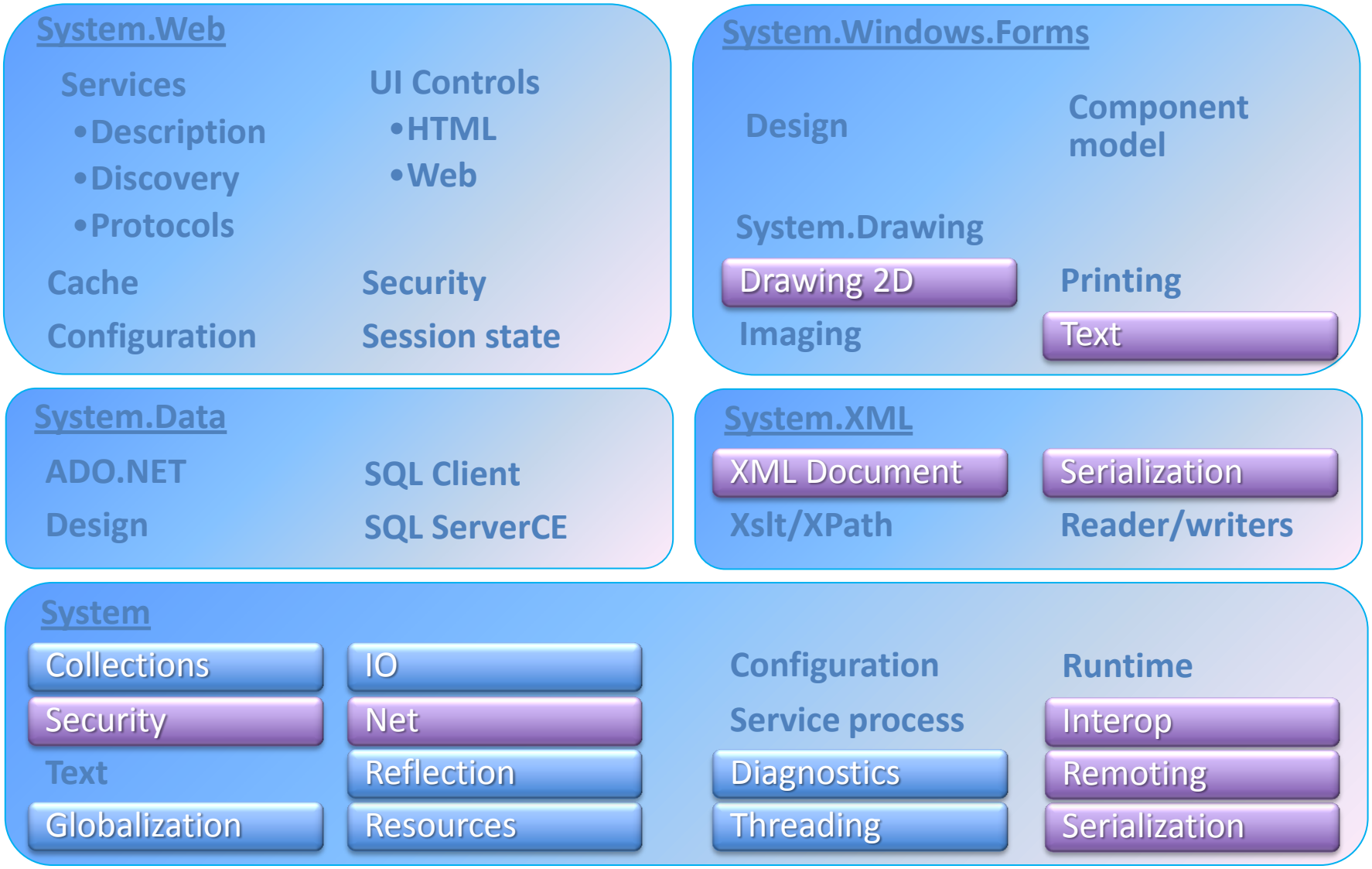
.NET Compact Framework

University of Hull



.NET Micro Framework

UNIVERSITY OF Hull



The Missing Operating System

- A device powered by the .NET Micro Framework does not have/need an operating system
- Programs execute directly on the hardware using a "Bootable Run Time System"
  - Simplifies deployment
  - Reduces the demands on the target platform
- Programs still have access to many parts of the .NET API
  - Provided by the run-time system rather than an operating system

How it works

- C# source is compiled to Intermediate Language (MSIL)
- The MSIL is downloaded in a compressed form and interpreted in the Micro Framework device
- As far as the code is concerned it is running inside a standard .NET Assembly
- The program runs as soon as the device is powered up

The .NET Framework Scope

- Not all .NET devices support all the components of the full .NET Framework
  - This is quite sensible given the limitations of the platforms and the needs of the solutions
- However, all the fundamentals of .NET are present on all of the platforms
  - And they are used in exactly the same way in your applications
  - And there are special purpose libraries

Objects and Hardware

- Input and output connections are managed by objects
- The output port object provides a method you can use to set the state of the pin
- You can also create ports which generate interrupts when they are triggered
  - These make use of delegates to despatch the interrupt event
- There are built in hardware abstractions for RS232, I2C, SPI and LCD

Reading an Input Pin

```
// Map the pin to the software
Cpu.Pin switchPin = Cpu.Pin.GPIO_Pin2;

// Create a port connected to a pin
InputPort switchInput = new InputPort(
        switchPin, //pin
        false,     //no glitch filter
        Port.ResistorMode.PullUp);

// Use the pin
If (switchInput.Read())
{
    Debug.Print("input high");
}
```

Graphical Displays

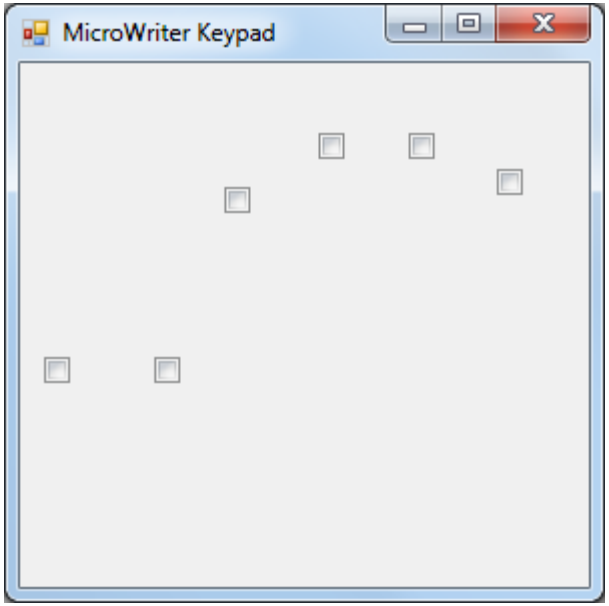
- The .NET Micro Framework has comprehensive graphical output facilities for driving device displays
- The facilities are provided at two levels:
  - Bitmap class provides a range of simple drawing facilities (shapes, images, text)
  - Windows Presentation Foundation (WPF) inspired set of display elements that can be used to create complex displays

The Emulator

- A program that interprets the Micro Framework code
- Provides very comprehensive debugging support
- A set of standard component emulations are included:
  - LCD display
  - General Purpose Input/Output
  - Serial port
  - I2C port

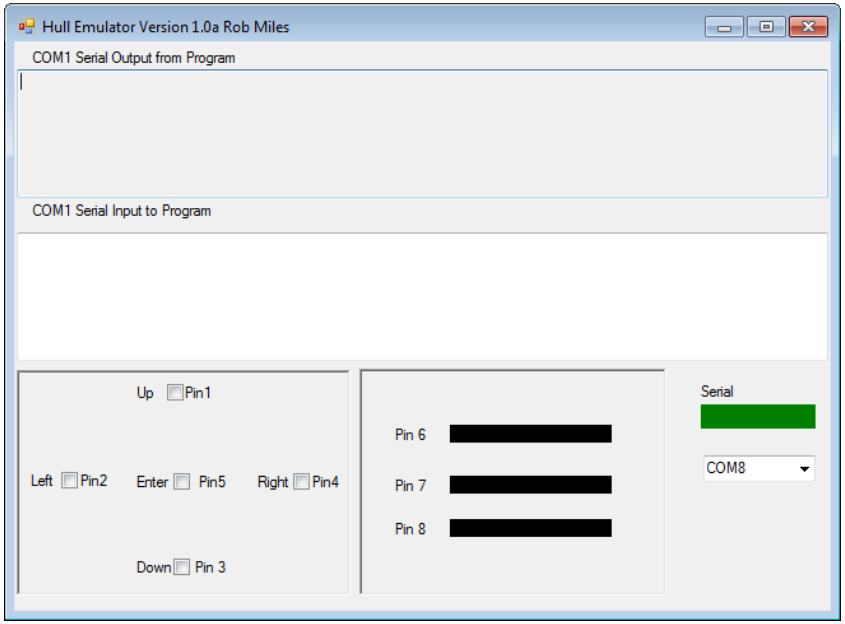
Extending the Emulator

- You can also create your own component emulations which can be integrated into your environment
- All this is done in C# by extending the existing emulator and component classes in your project
- You can easily build up a library of device and component emulations which can be shared across projects



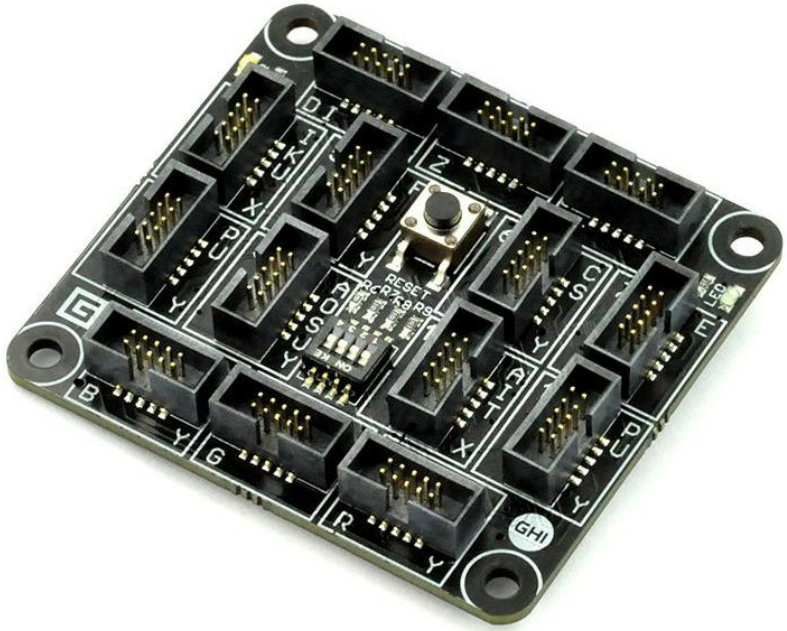
The Hull Emulator

- I have created a custom emulator for use at Hull
- It provides a set of input and output ports that you can use to test your programs
- It also provides a simulated serial port connection for testing programs that connect to other systems



Gadgeteer

- This is a Microsoft Gadgeteer device
- It is powered by the .NET Micro Framework but provides a much easier way to interface devices
- This side has all the sockets

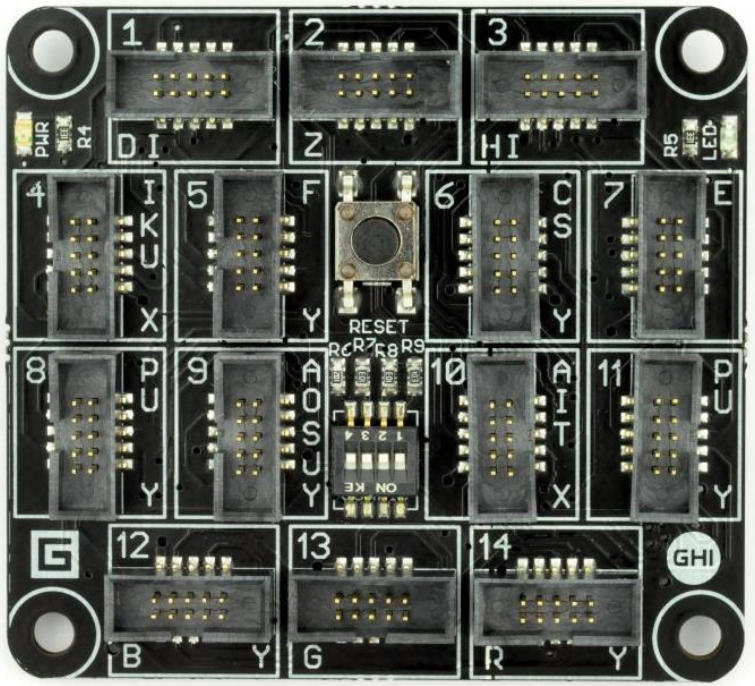


Computers for Gadgets

- This side has the processor
- You could create an embedded device on your own PC and then just add this processing element
- This makes the Gadgeteer devices great for prototyping

Plenty of connections

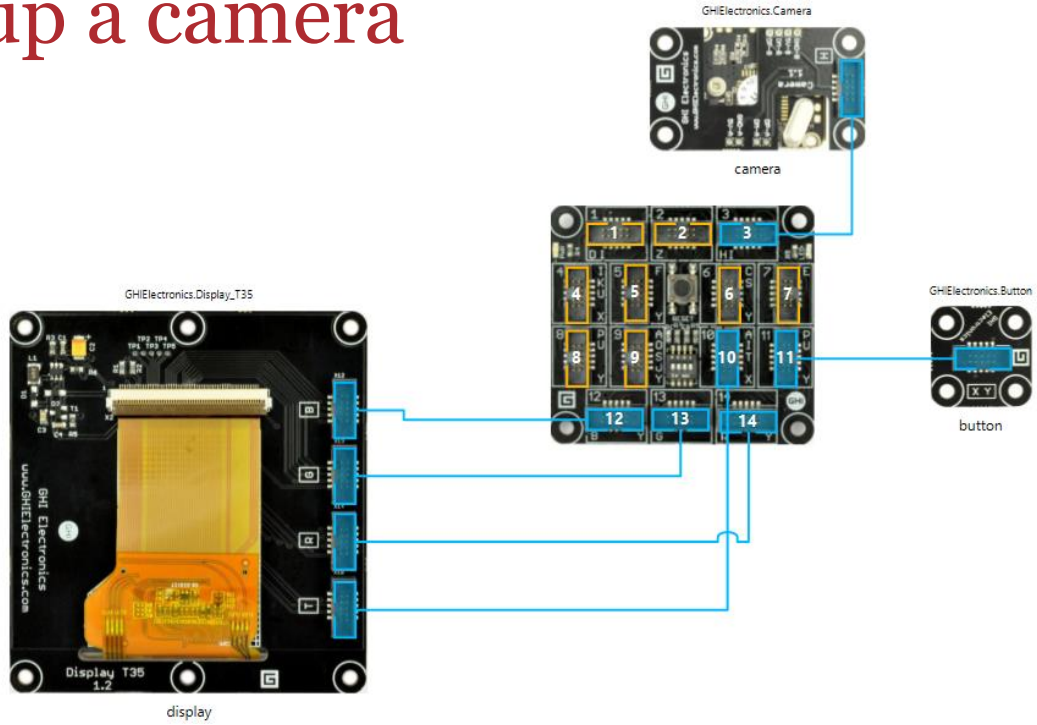
- We can connect a device to each of the sockets
- Power goes into socket number 1
- Some devices use several sockets
- Each socket has a



So, how do we make Gadgets?

- To make a gadget we have to program the processor (the bit in the middle) with the software that will make it do what we want
- We also have to work out which devices fit into which sockets
- We use a tool called Visual Studio to do this
- It has a Gadget design surface that connects the components together

Wiring up a camera



- These are the connections for a simple digital camera
- It uses an LCD panel, a camera sensor and a button
- Once we have the sensors connected we can write the code

Responding to Events

```
button.ButtonPressed +=
    new Button.ButtonEventHandler(button_ButtonPressed);

camera.PictureCaptured +=
    new Camera.PictureCapturedEventHandler(camera_PictureCaptured);
```

- These two program statements "wire" methods up to events that the components generate
  - The first one fires a method when a button is pressed
  - The second one fires a method when the camera has a picture ready

Responding to the button event

```
void button_ButtonPressed(Button sender, Button.ButtonState state)
{
    camera.TakePicture();
}
```

- This is the C# code that runs when the user presses the button
- It just asks the camera to take a picture
- The camera is represented by a software object that contains methods a program can use to ask it to do things

Responding when the picture is ready

```
void camera_PictureCaptured(Camera sender, GT.Picture picture)
{
    display.SimpleGraphics.DisplayImage(picture,0,0);
}
```

- This is the C# code that runs when the camera has a picture ready
- The camera will run this method when the picture arrives
- The method just takes the picture that was received and puts it on the screen

Is that all there is?

- This is all the code we need to write to create a digital camera
- We could change the software
  - Make a self timer that waits a few seconds before taking the picture
  - Make the camera take a sequence of shots rather than just one, then let you pick the best
  - Turn the picture into black and white
  - Combine two pictures so you can be in the same scene twice

demo

“Make a Camera”

Demo 1: Creating a Camera from Scratch

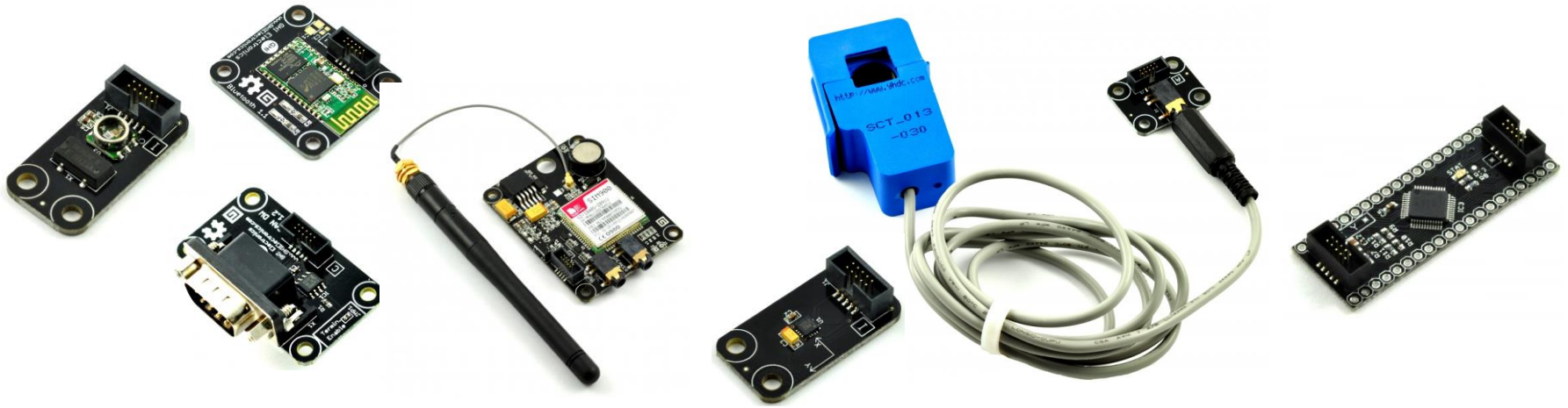
Gadgeteer Devices

- The camera used a number of devices
  - The camera
  - The button
  - The LCD panel
- We could add others
  - Add an SD card to save the picture
  - Add a network connection to put the picture on the web
- There are lots of devices we can use?

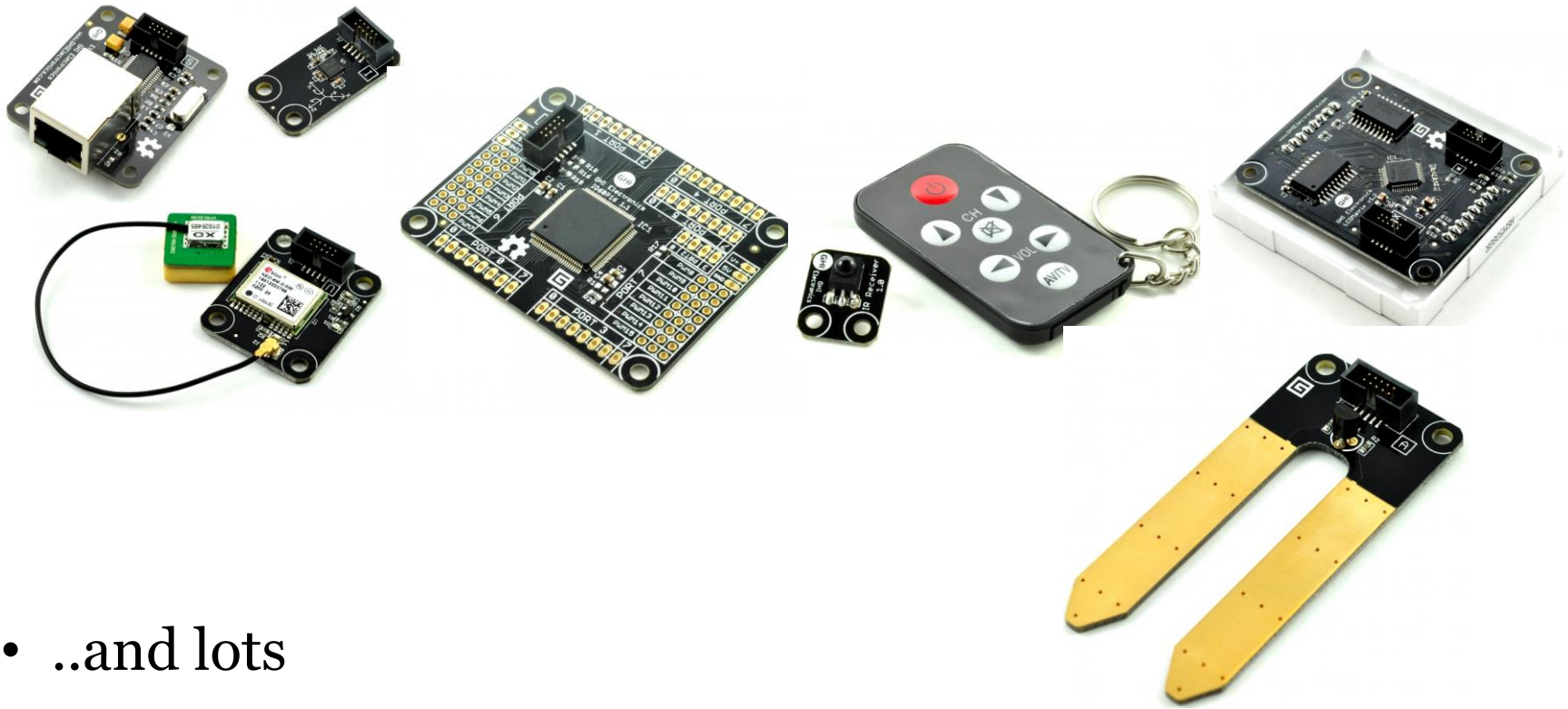
What do we mean by device?

- There are lots of devices

What do we mean by device?

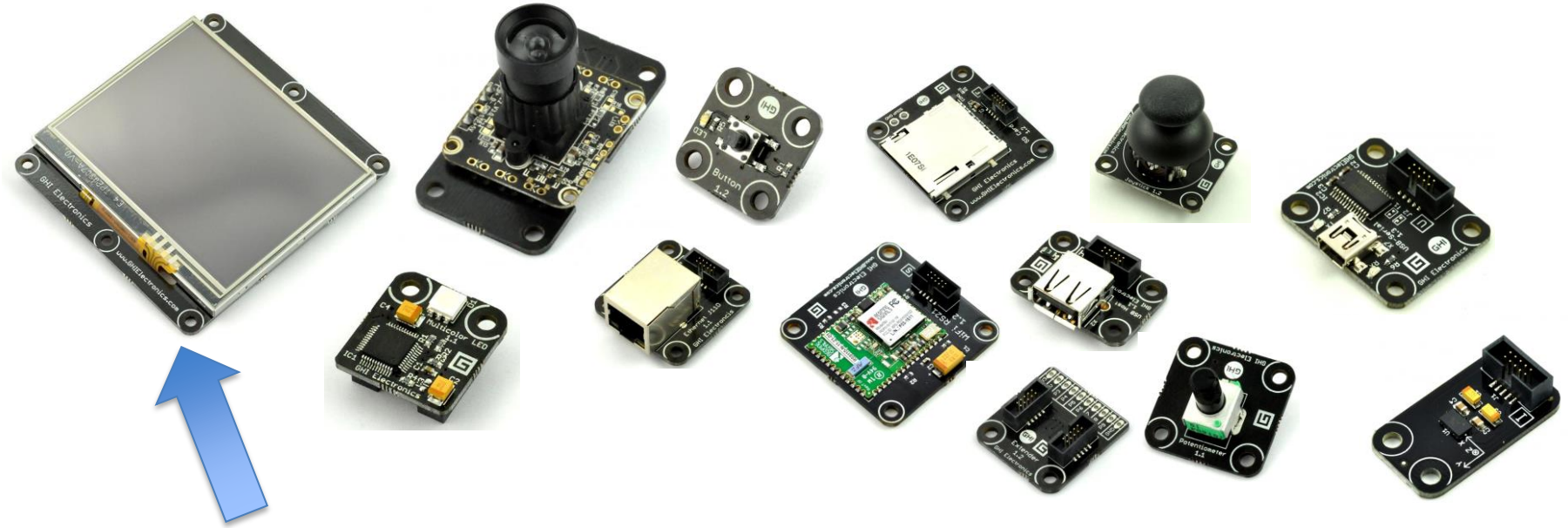


- ..and lots



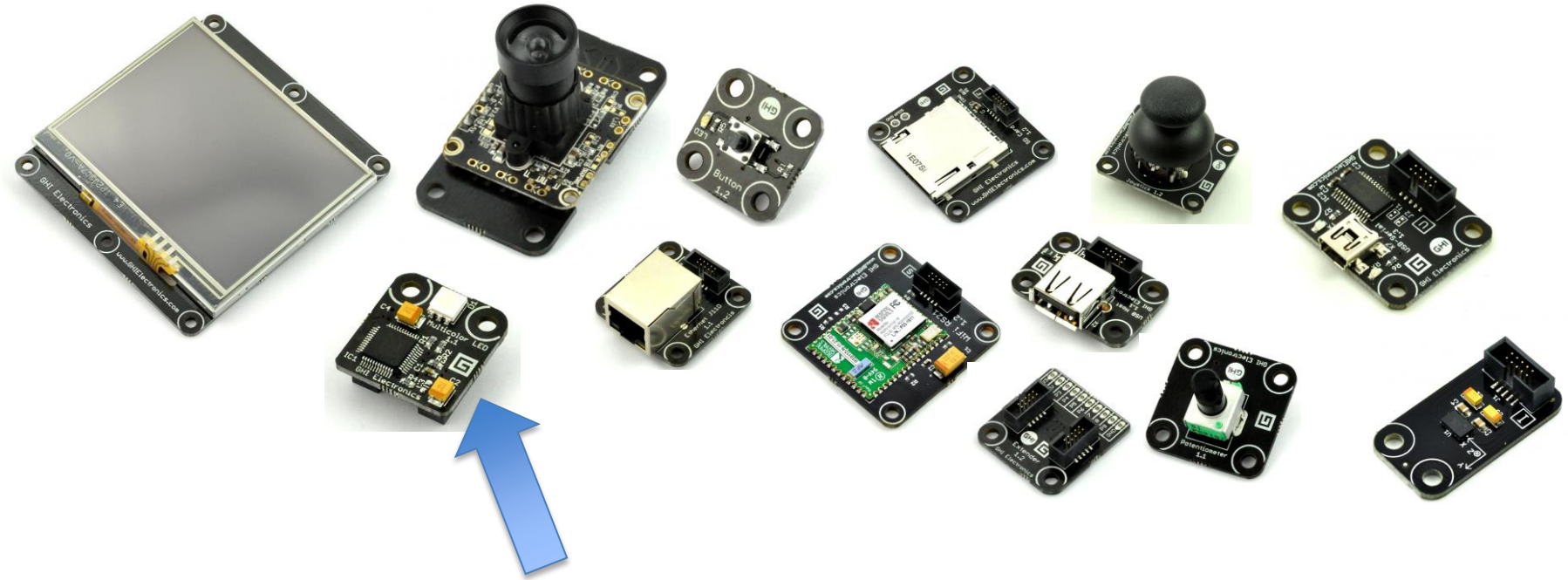
- ..and lots

What do we mean by device?



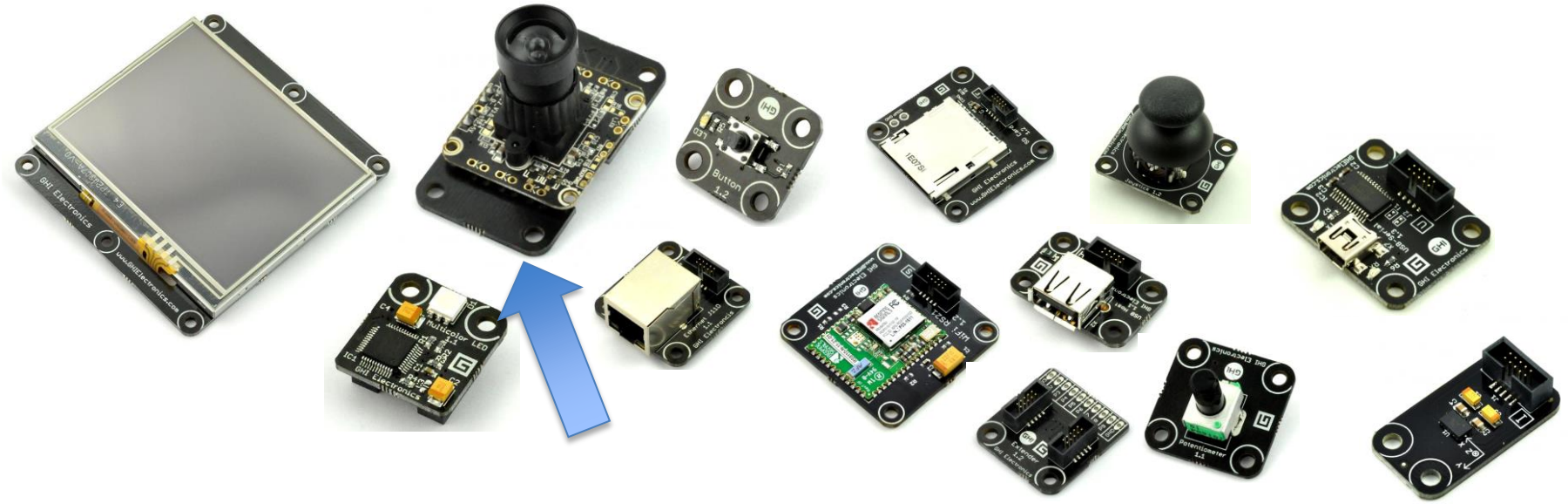
- LCD panel with touch screen input
  - A gadget can display text and graphics
  - Not all gadgets need a display though, many are "headless"

What do we mean by device?



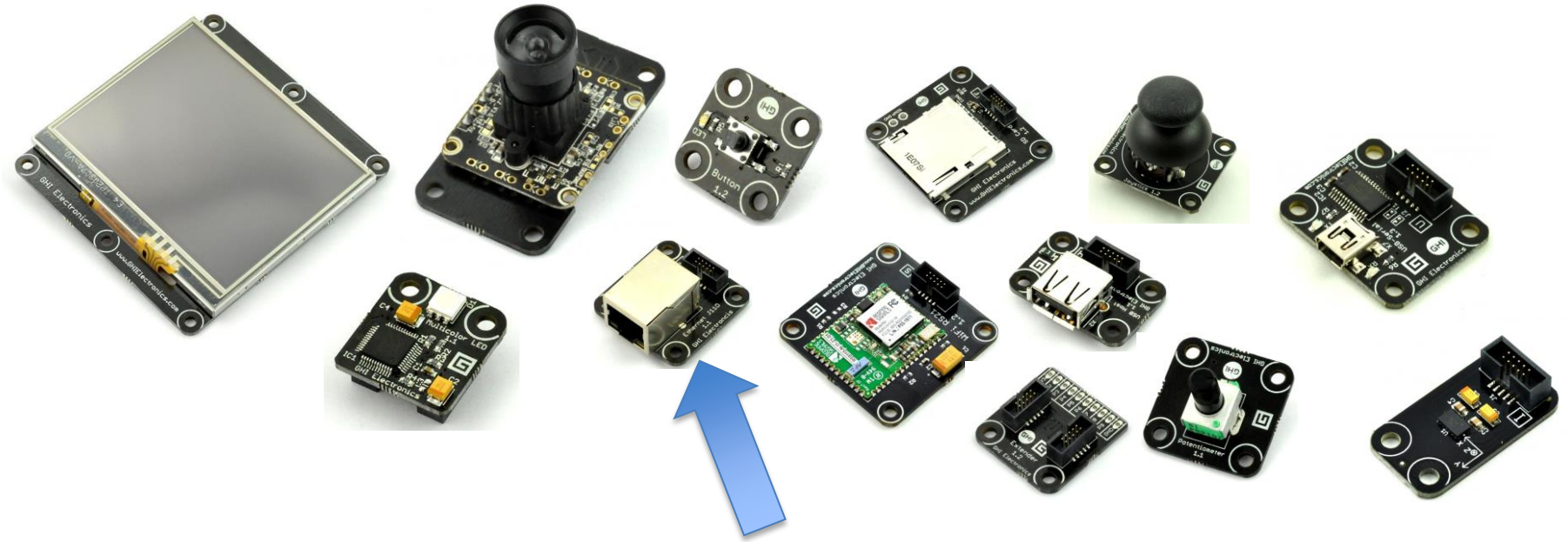
- Coloured LED
  - The LED can be programmed change colour and flash automatically

What do we mean by device?



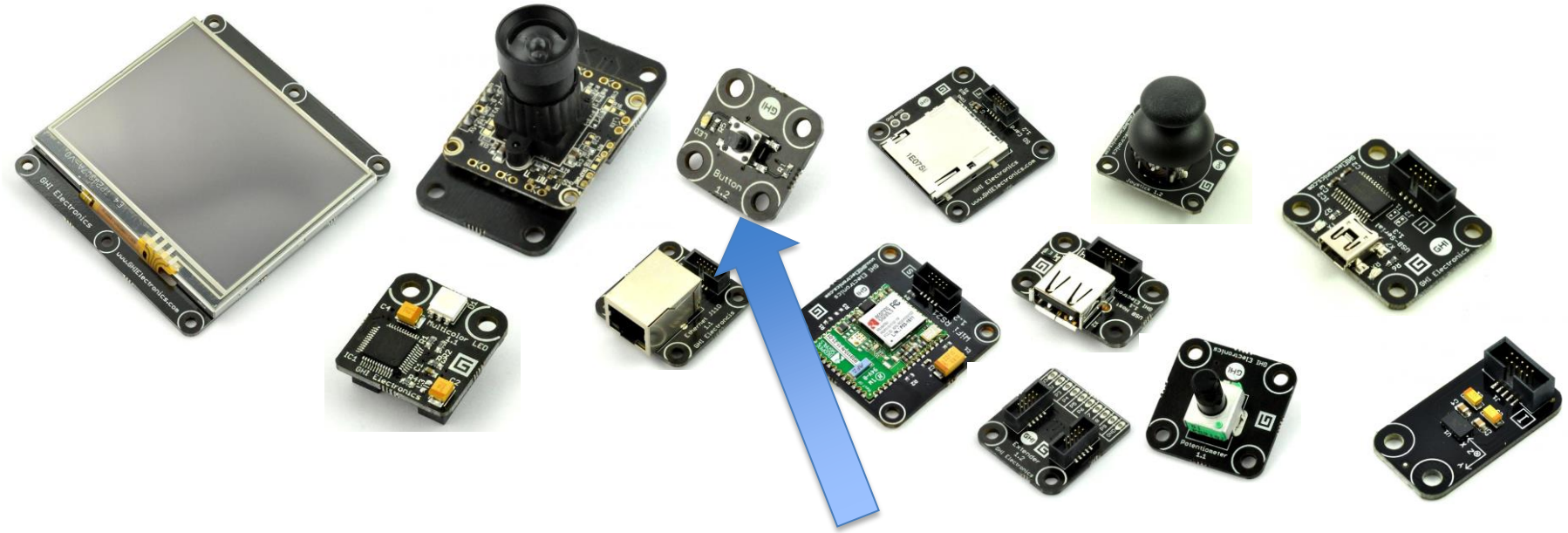
- Video Camera
  - A gadget can take pictures of its surroundings

What do we mean by device?



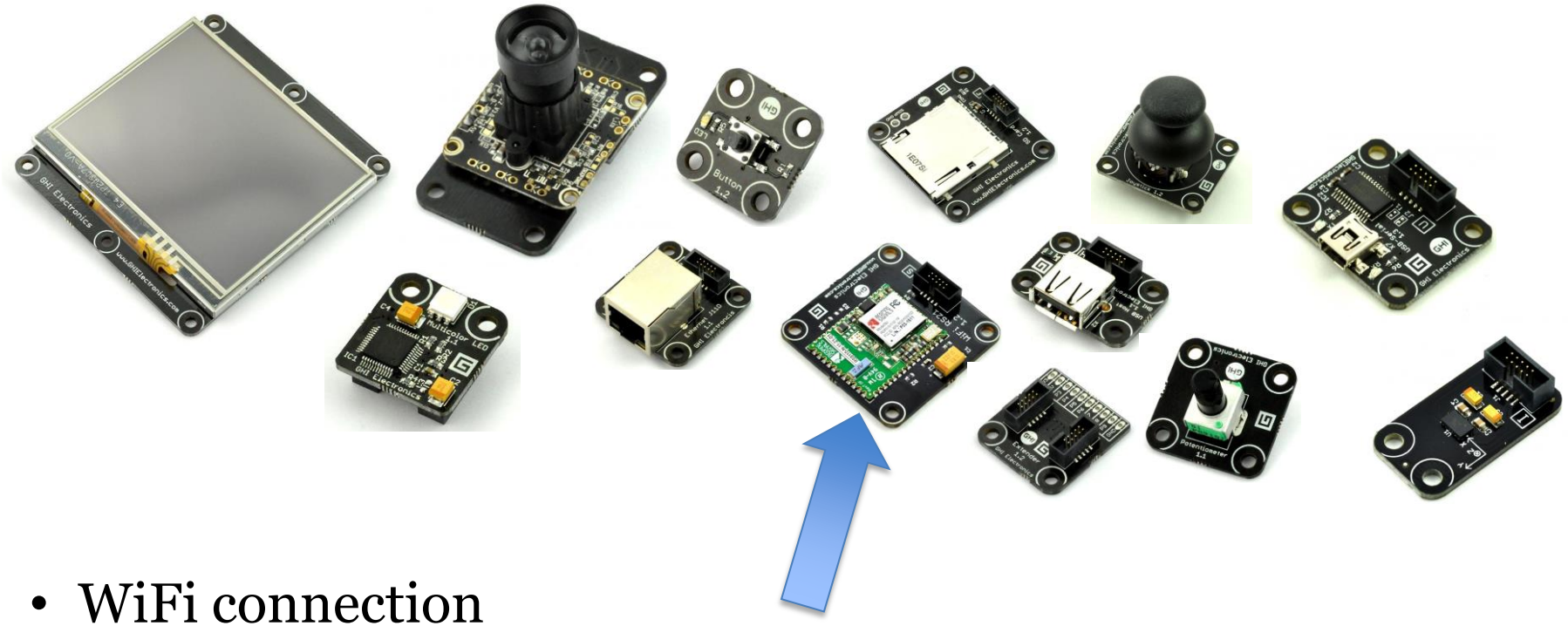
- Network Connection
  - A gadget can connect to a wired network and send and receive data

What do we mean by device?



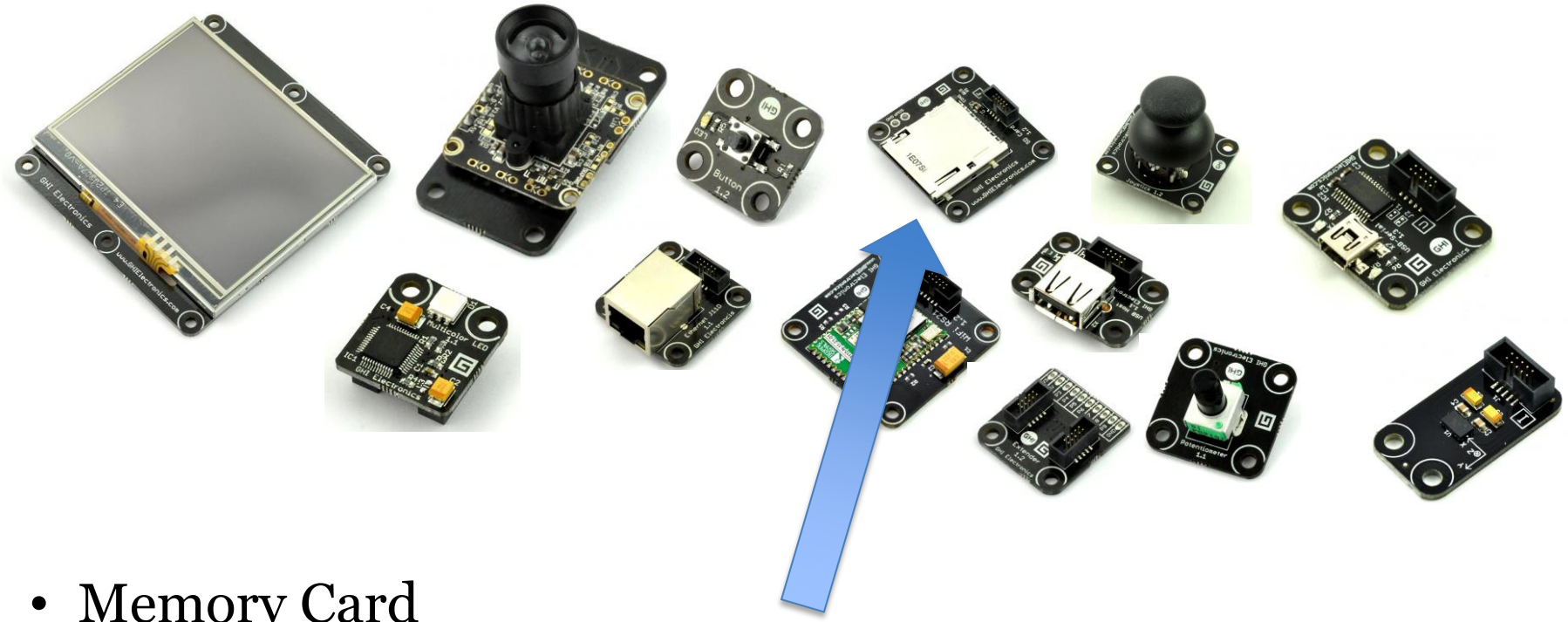
- Push button
  - The user can make things happen by pressing the button
  - This can start a program running inside the gadget

What do we mean by device?



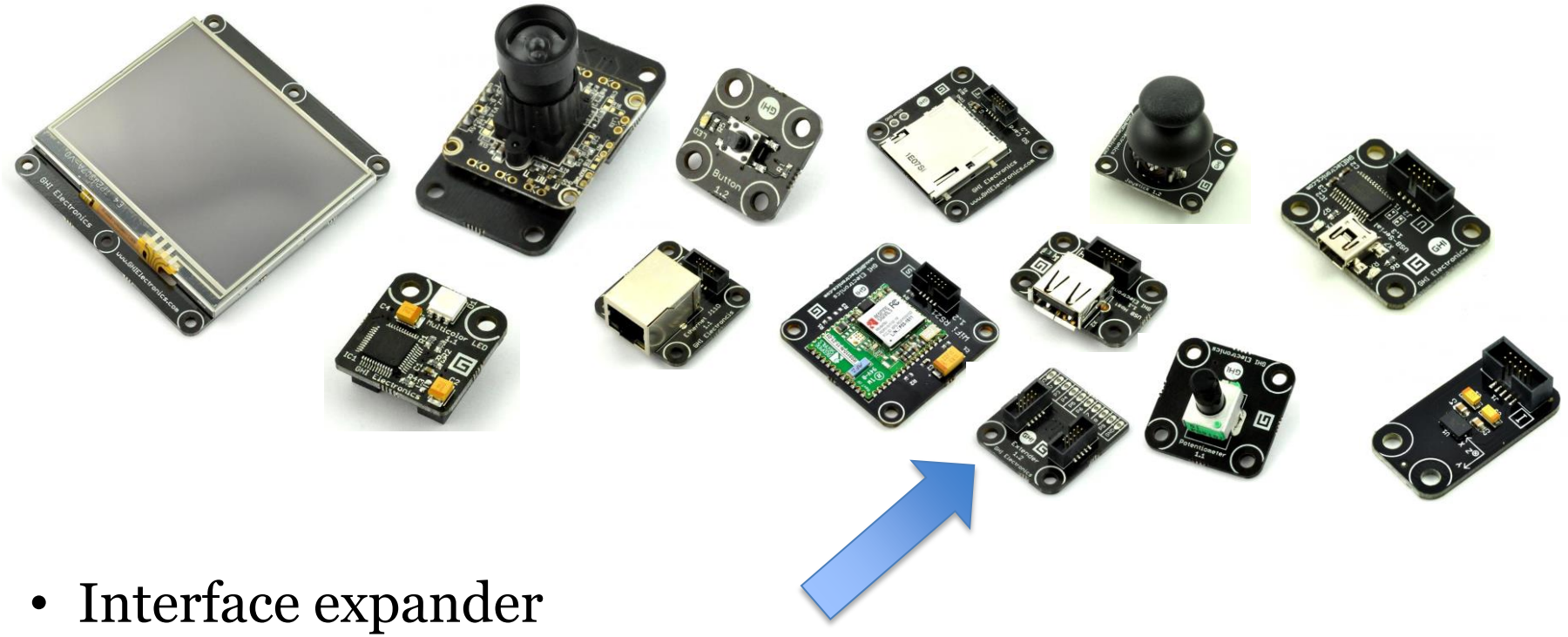
- WiFi connection
  - Gadgets can send and receive information over wireless networks

What do we mean by device?



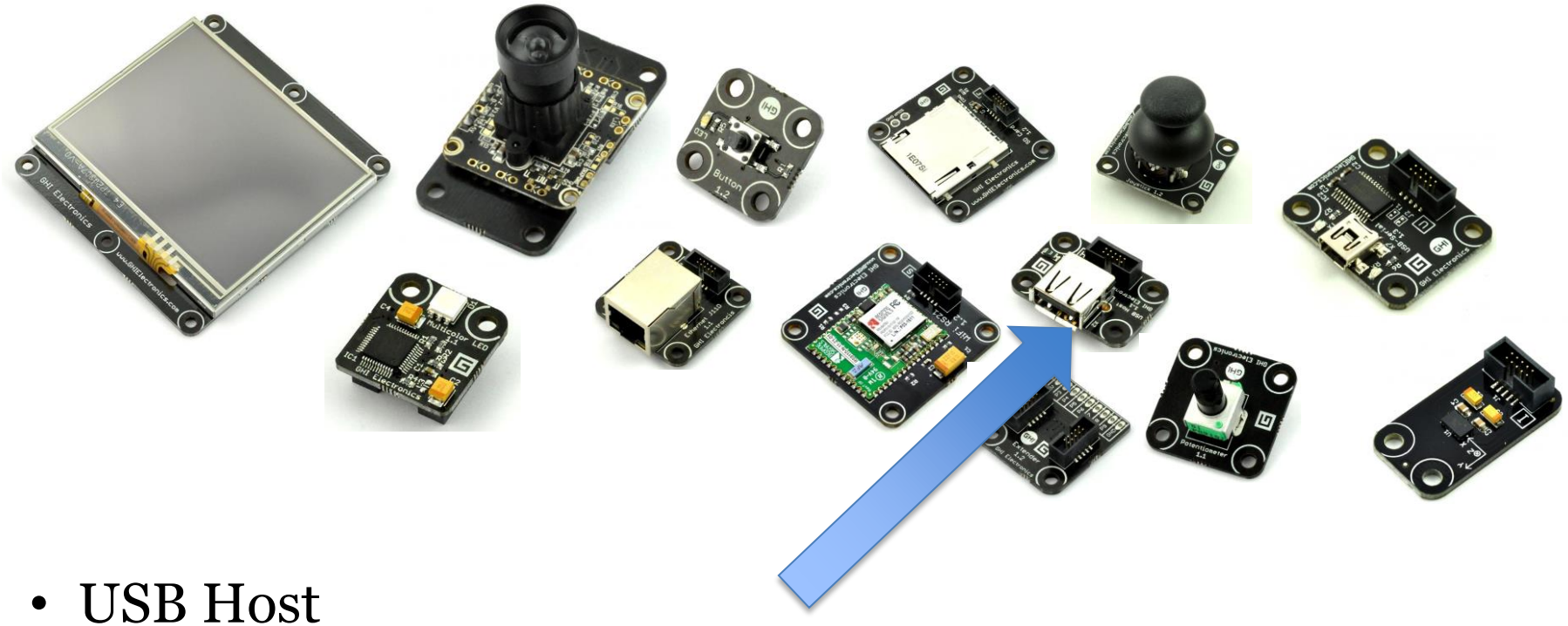
- Memory Card
  - Gadgets can store data in an SD card or read data back from it

What do we mean by device?



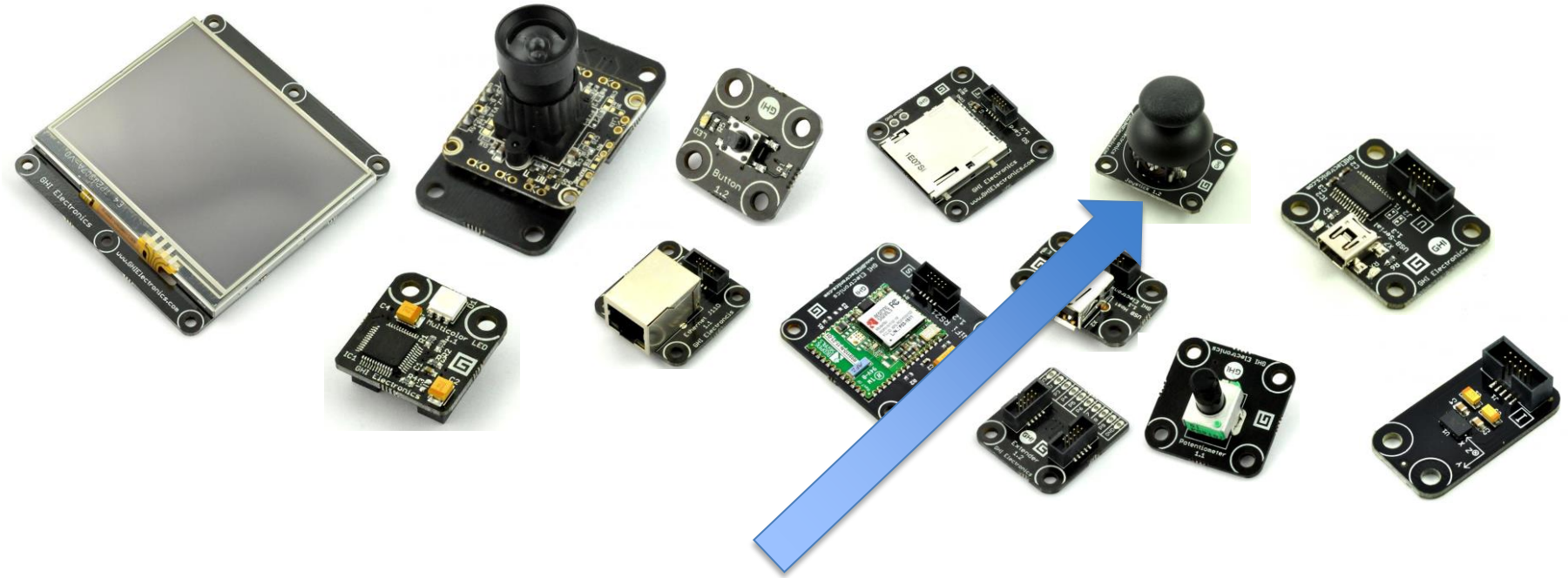
- Interface expander
  - Gadgets can connect directly to other devices
  - We will use this to connect a printer

What do we mean by device?



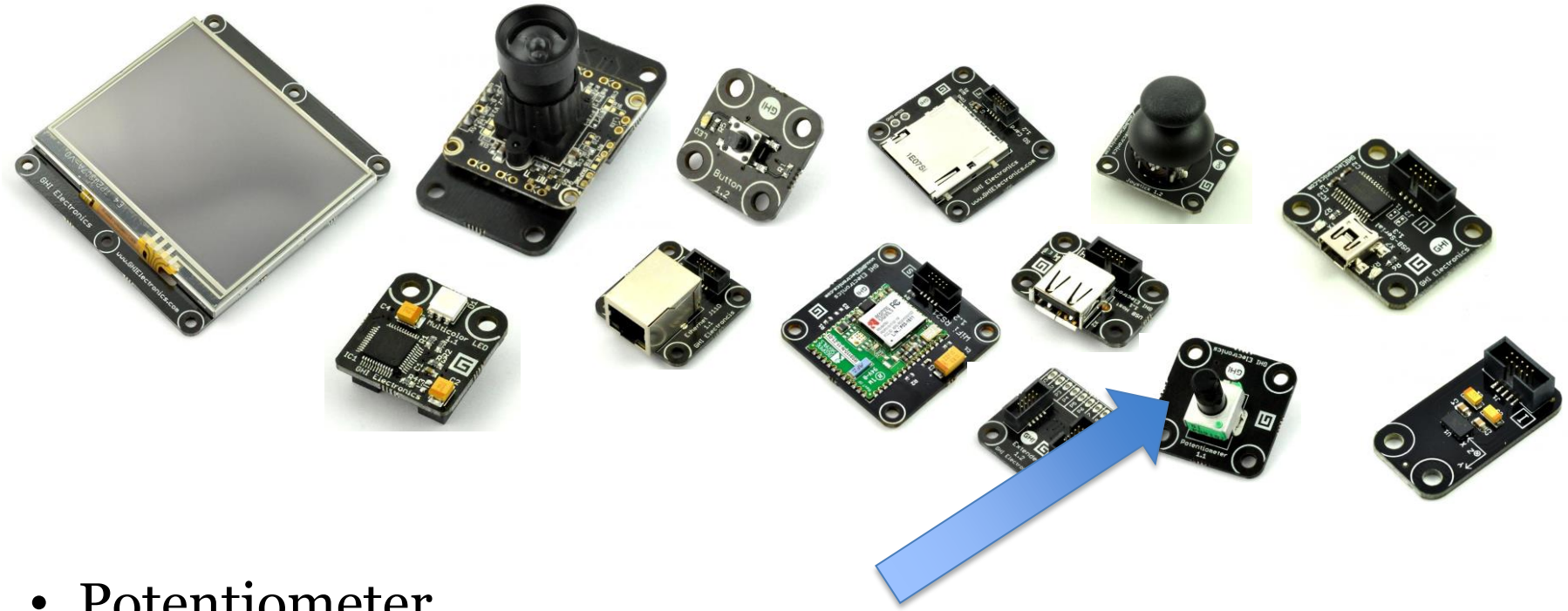
- USB Host
  - Gadgets can connect to lots of other peripherals, for example mice or keyboards

What do we mean by device?



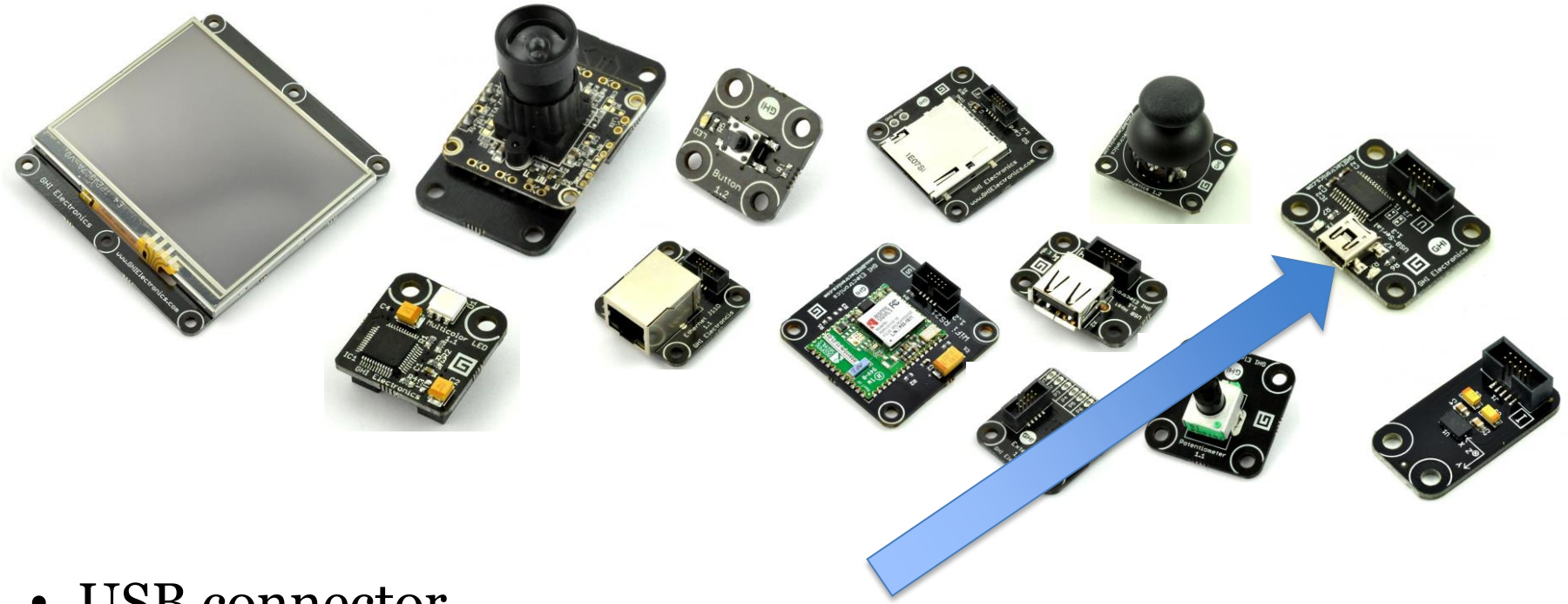
- Joystick
  - Users can control their gadgets using a joystick

What do we mean by device?



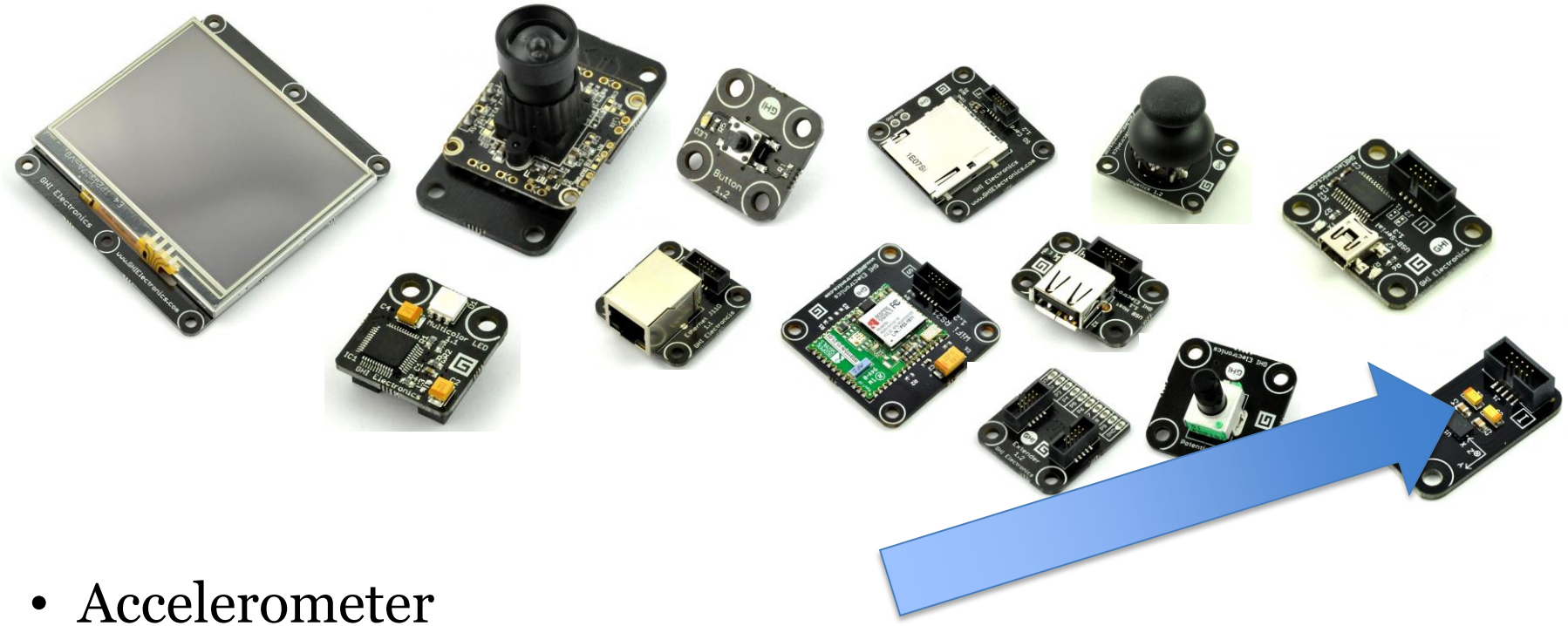
- Potentiometer
  - Users can control their gadgets turning a knob

What do we mean by device?



- USB connector
  - Links a Gadget to another computer

What do we mean by device?

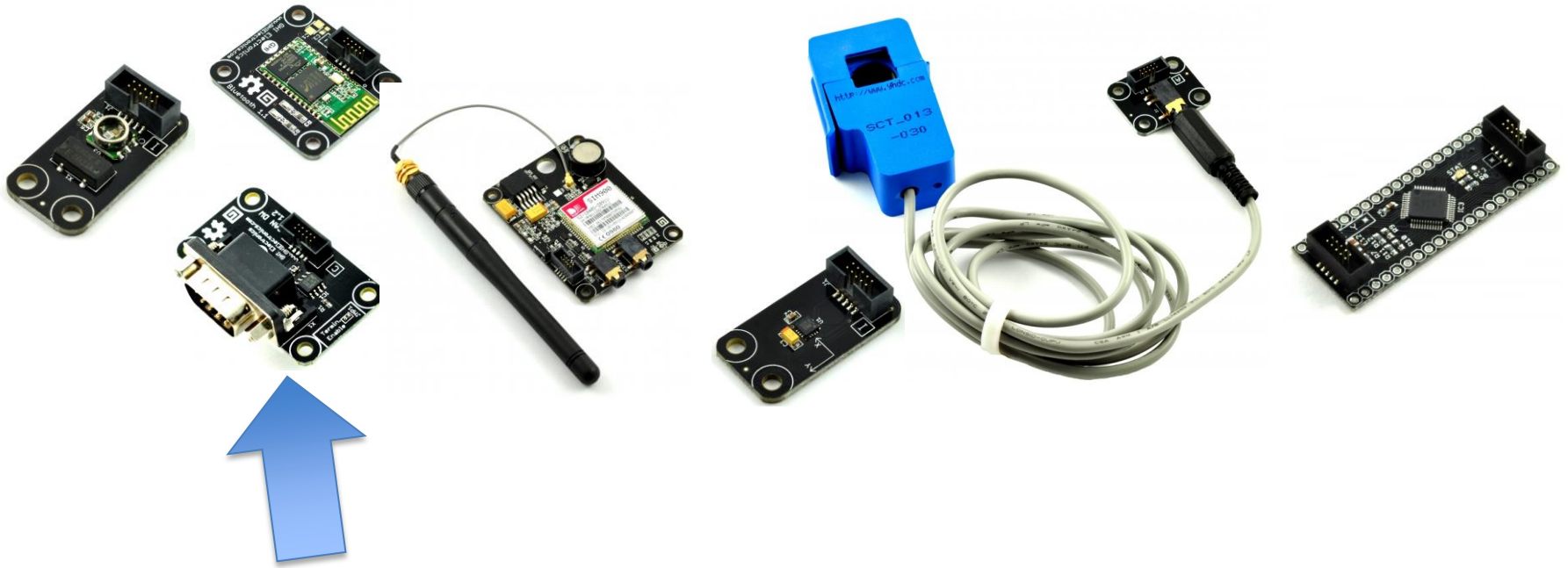


- Accelerometer
  - A Gadget can read the acceleration acting on it at any time

What do we mean by device?

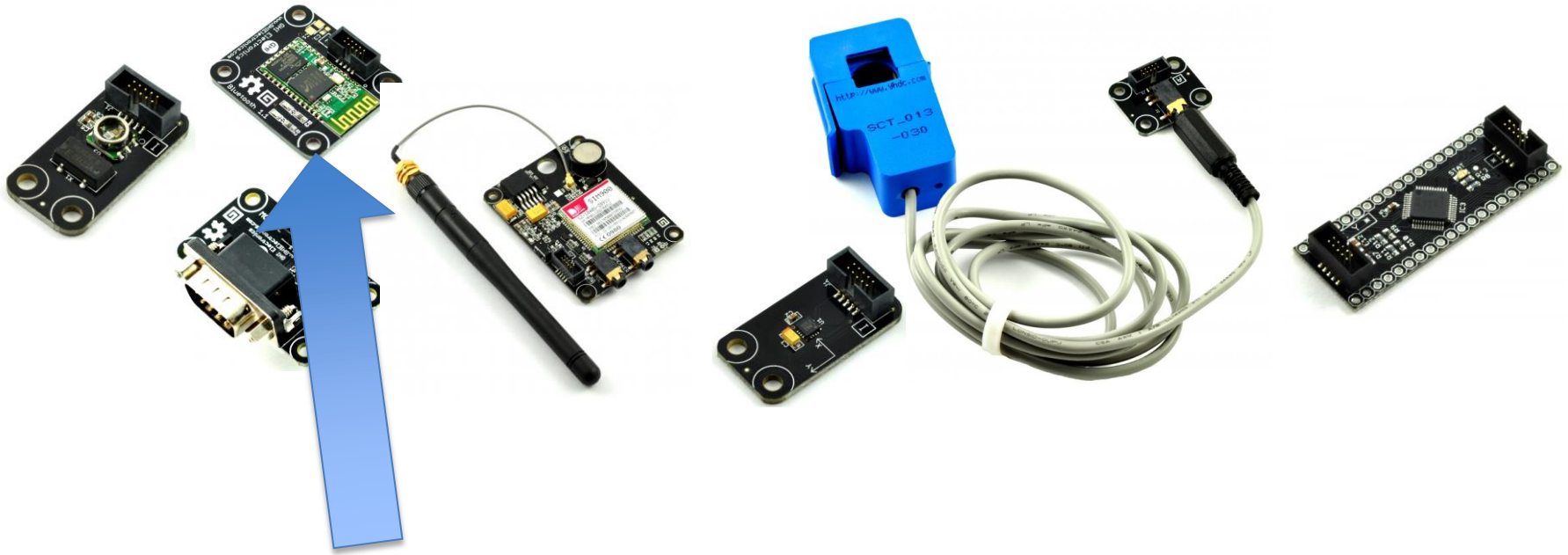
- Barometer
  - A Gadget can measure air pressure

What do we mean by device?



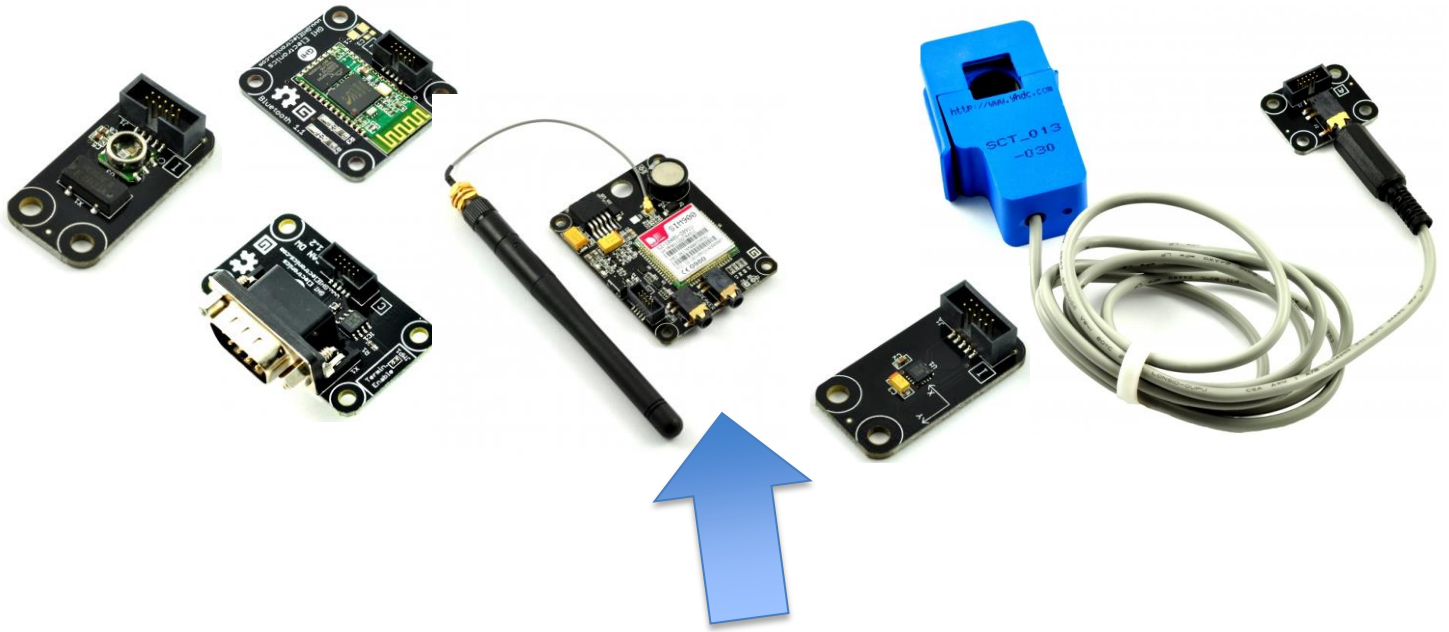
- CAN interface
  - A Gadget can connect to industrial machinery

What do we mean by device?



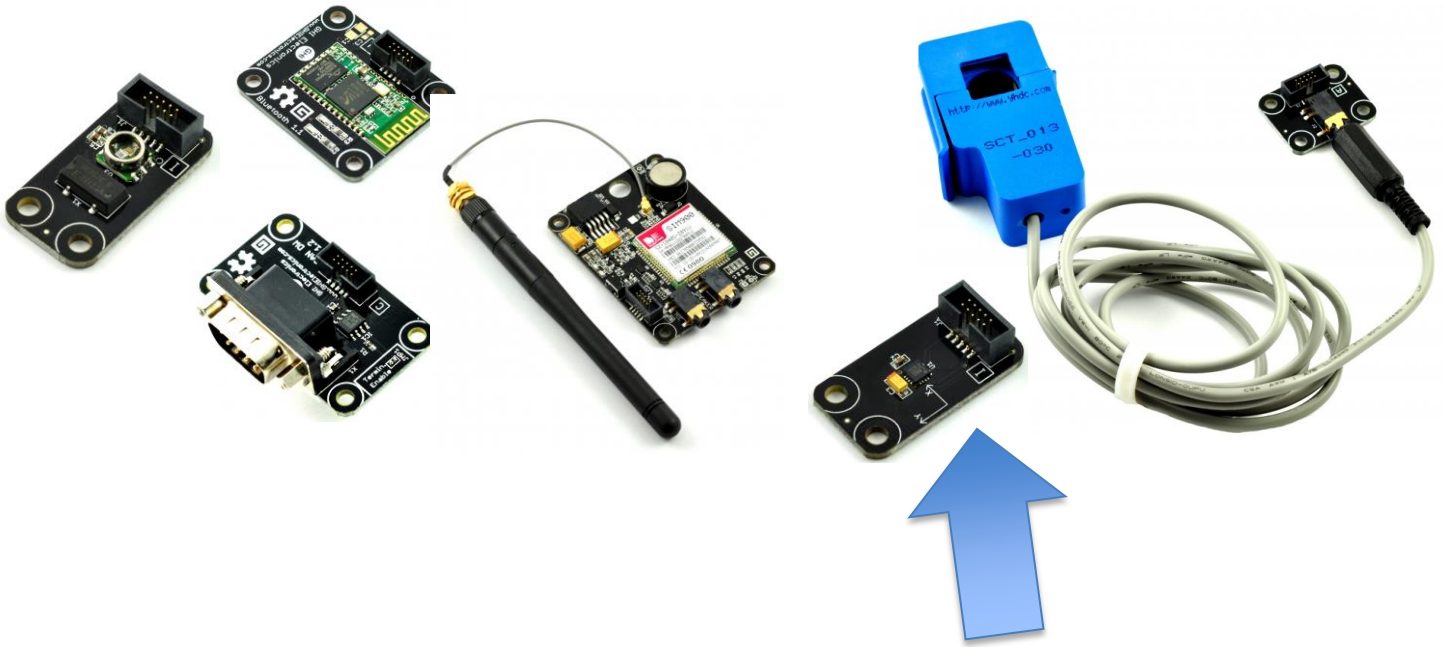
- Bluetooth interface
  - A Gadget can connect to all kinds of Bluetooth devices

What do we mean by device?



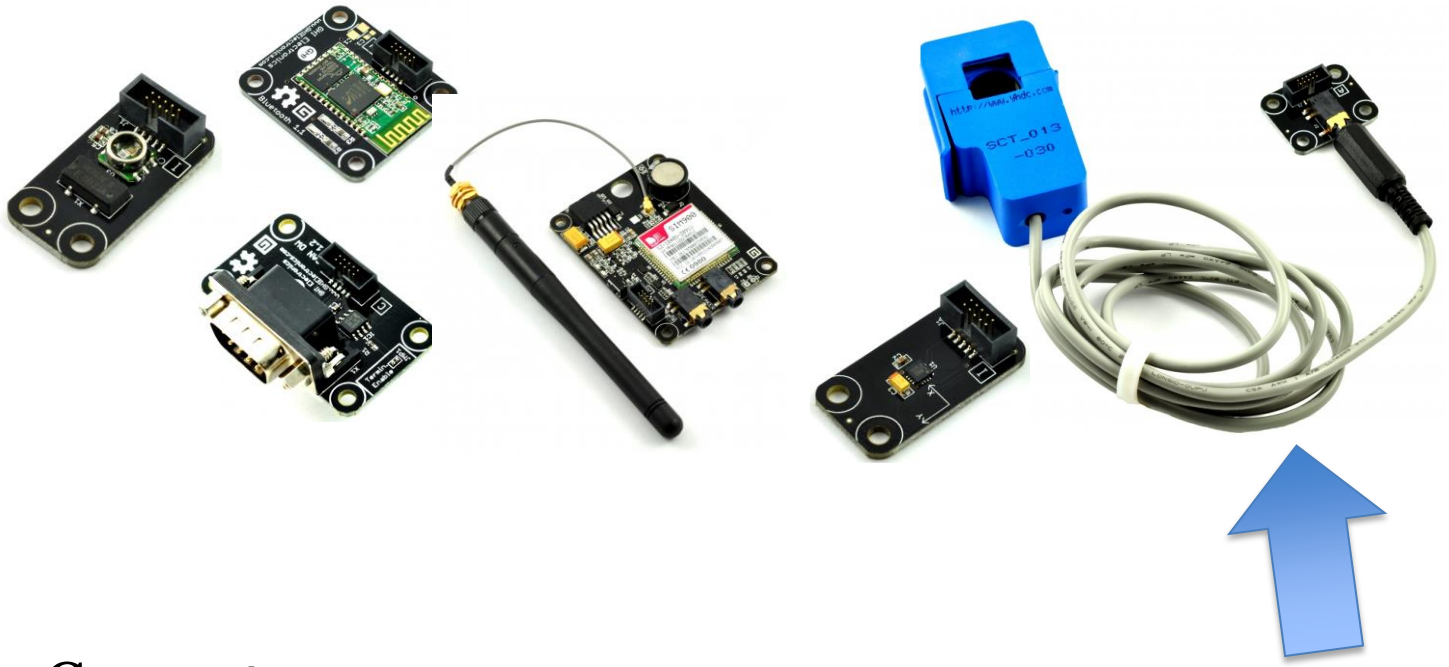
- Mobile Phone
  - A Gadget can send and receive data over the mobile phone network

What do we mean by device?



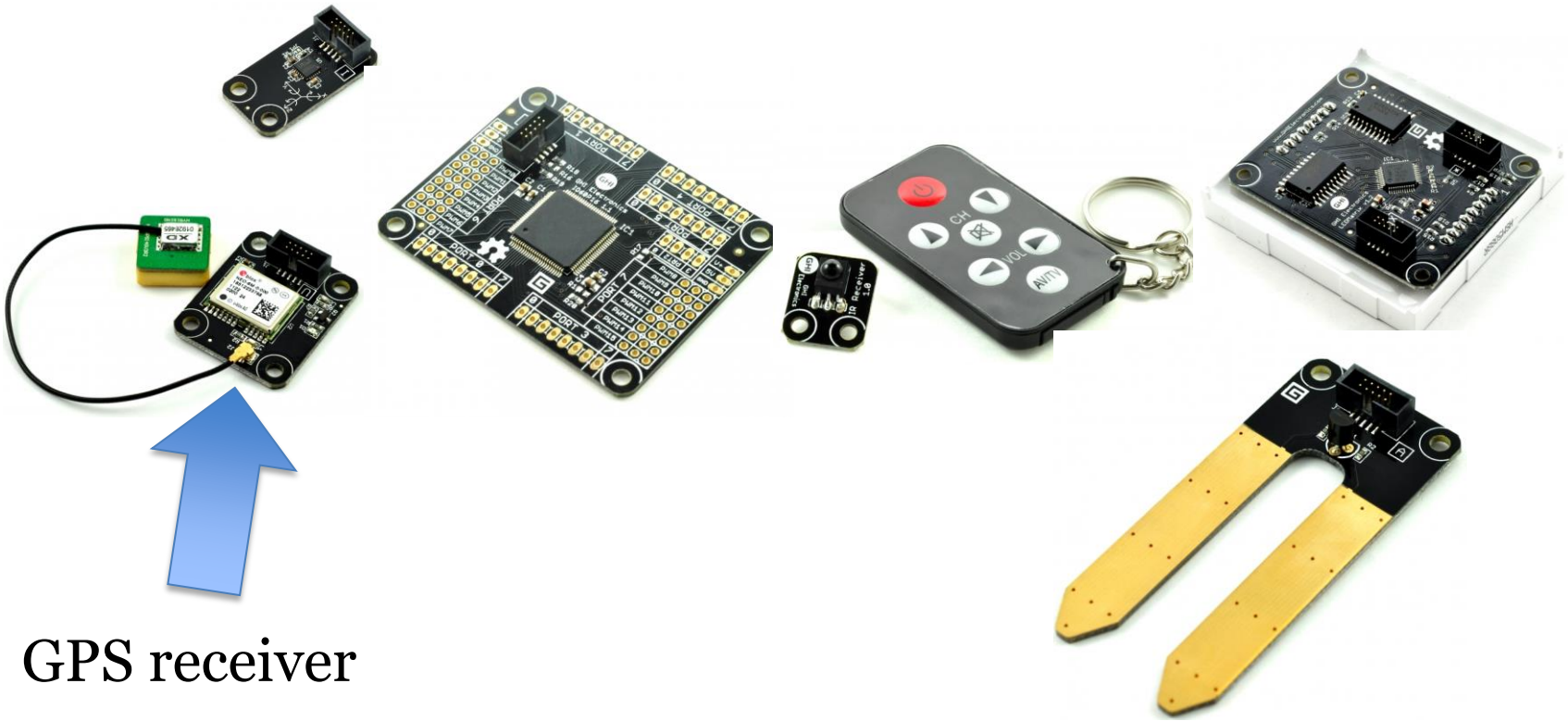
- Compass
  - A Gadget can work out which way is north

What do we mean by device?



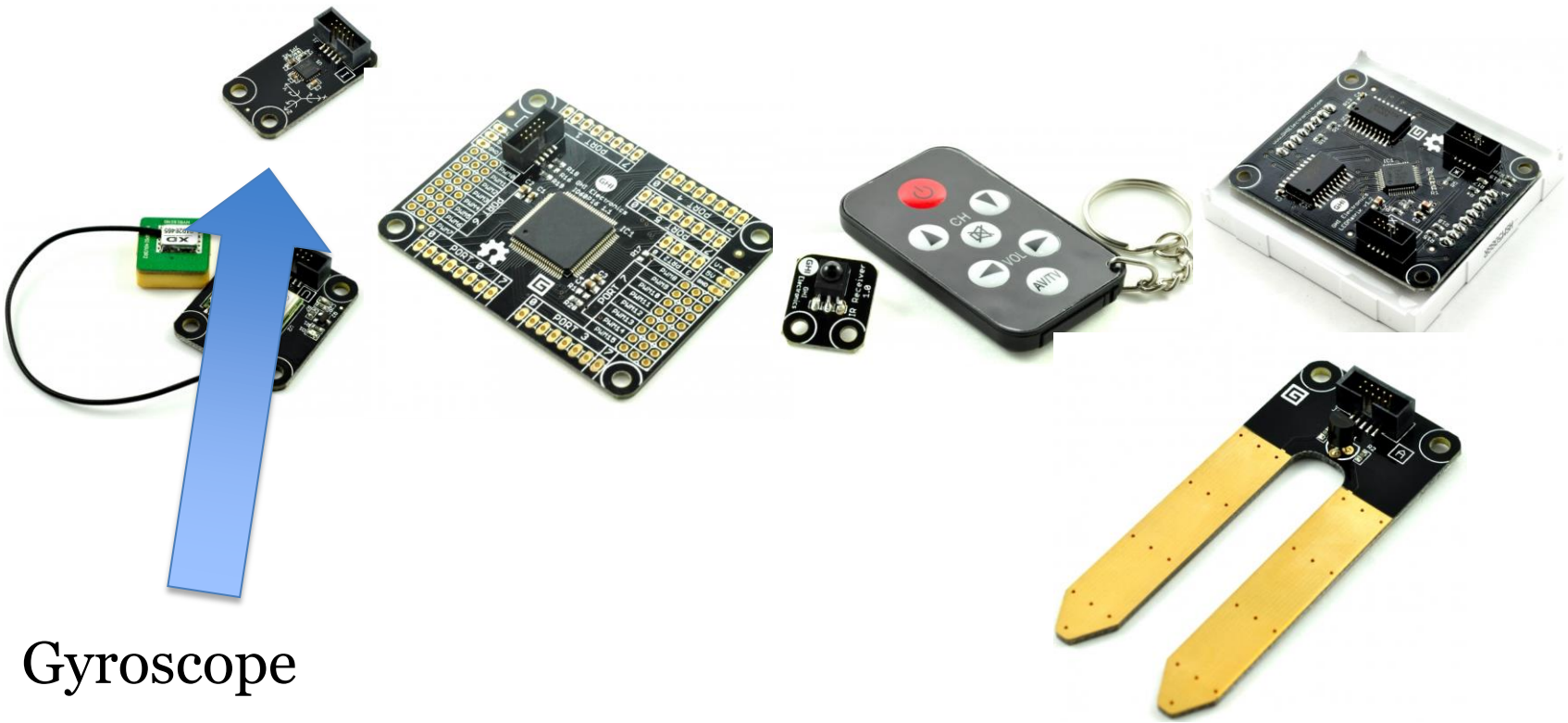
- Current sensor
  - A Gadget can discover the amount of power a circuit is using

What do we mean by device?



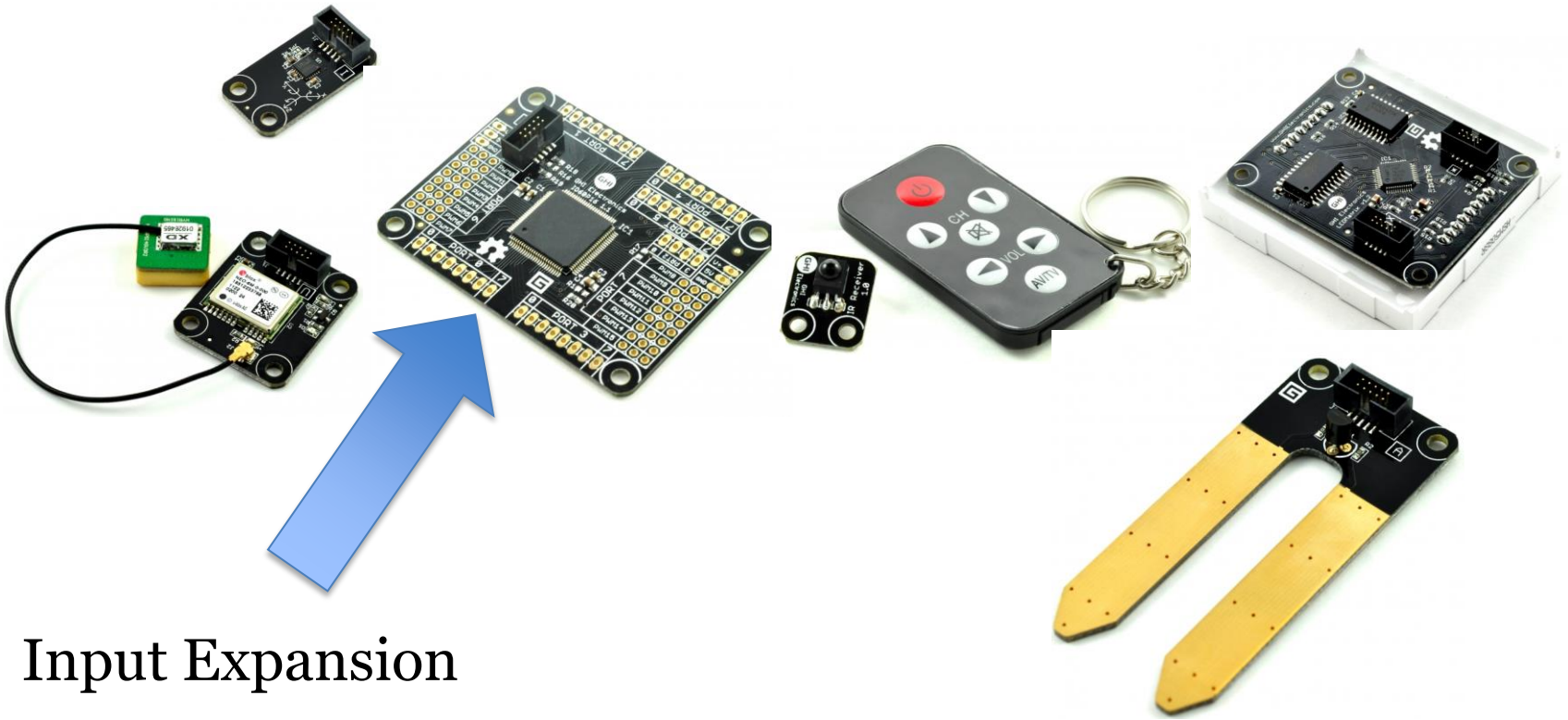
- GPS receiver
  - A gadget can tell where it is

What do we mean by device?



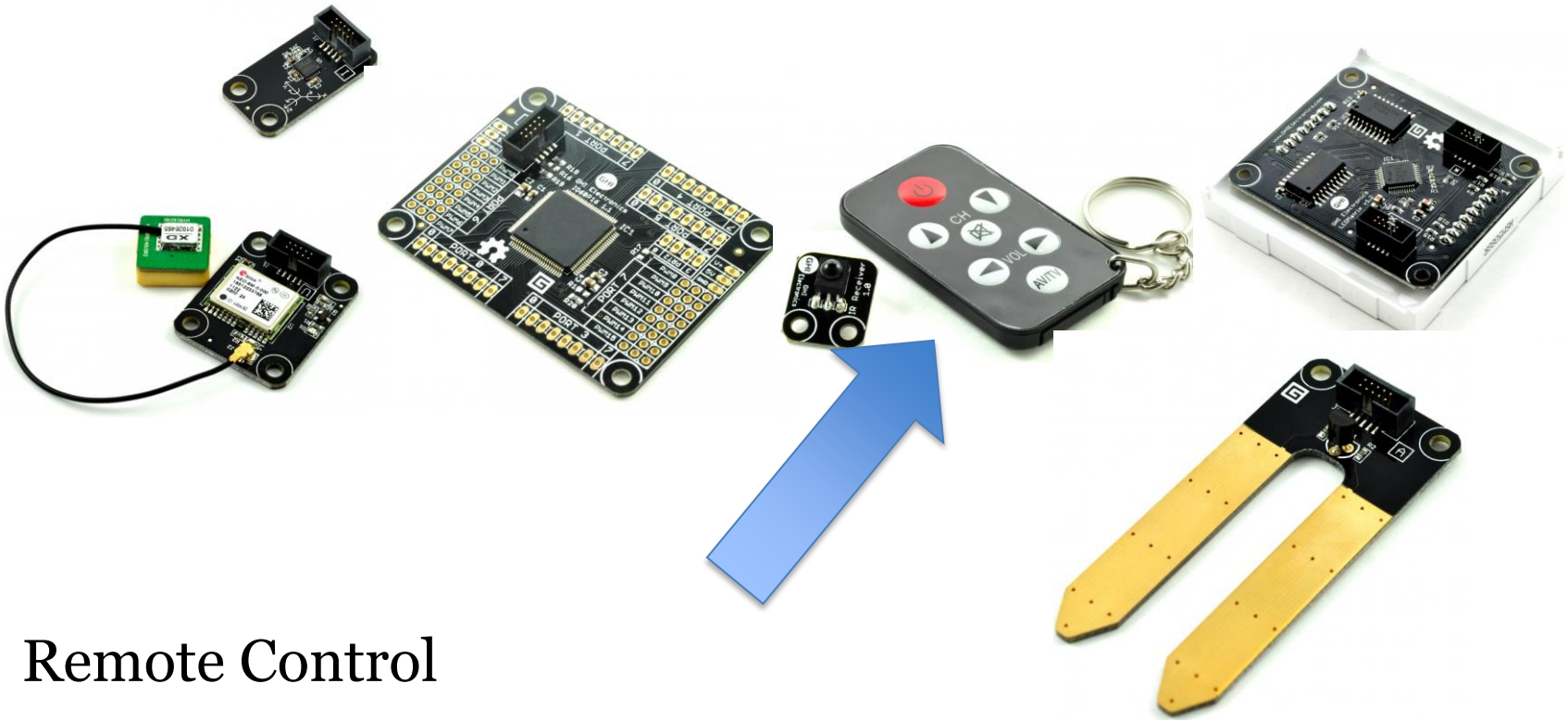
- Gyroscope
  - A gadget can tell where if it is turning

What do we mean by device?



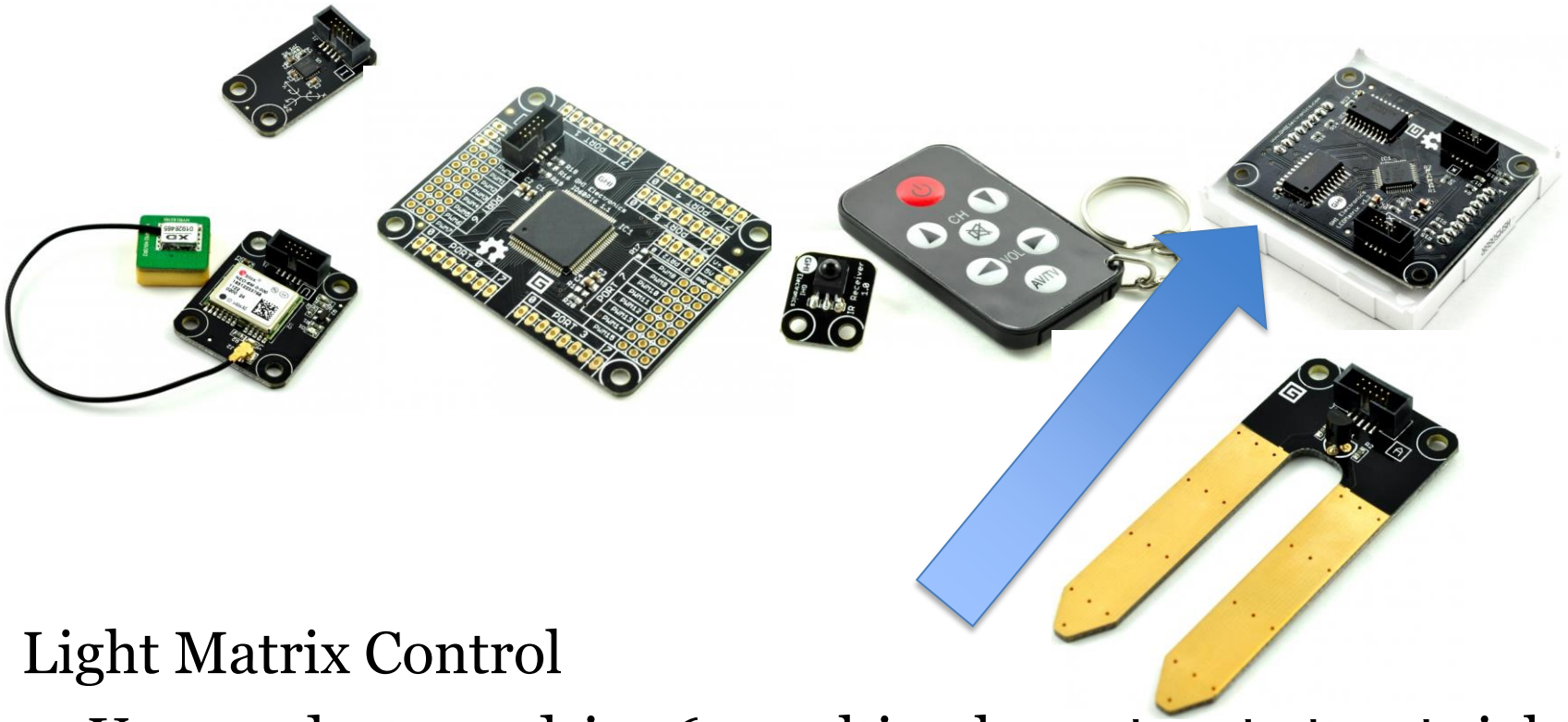
- Input Expansion
  - A gadget can control lots of other devices

What do we mean by device?



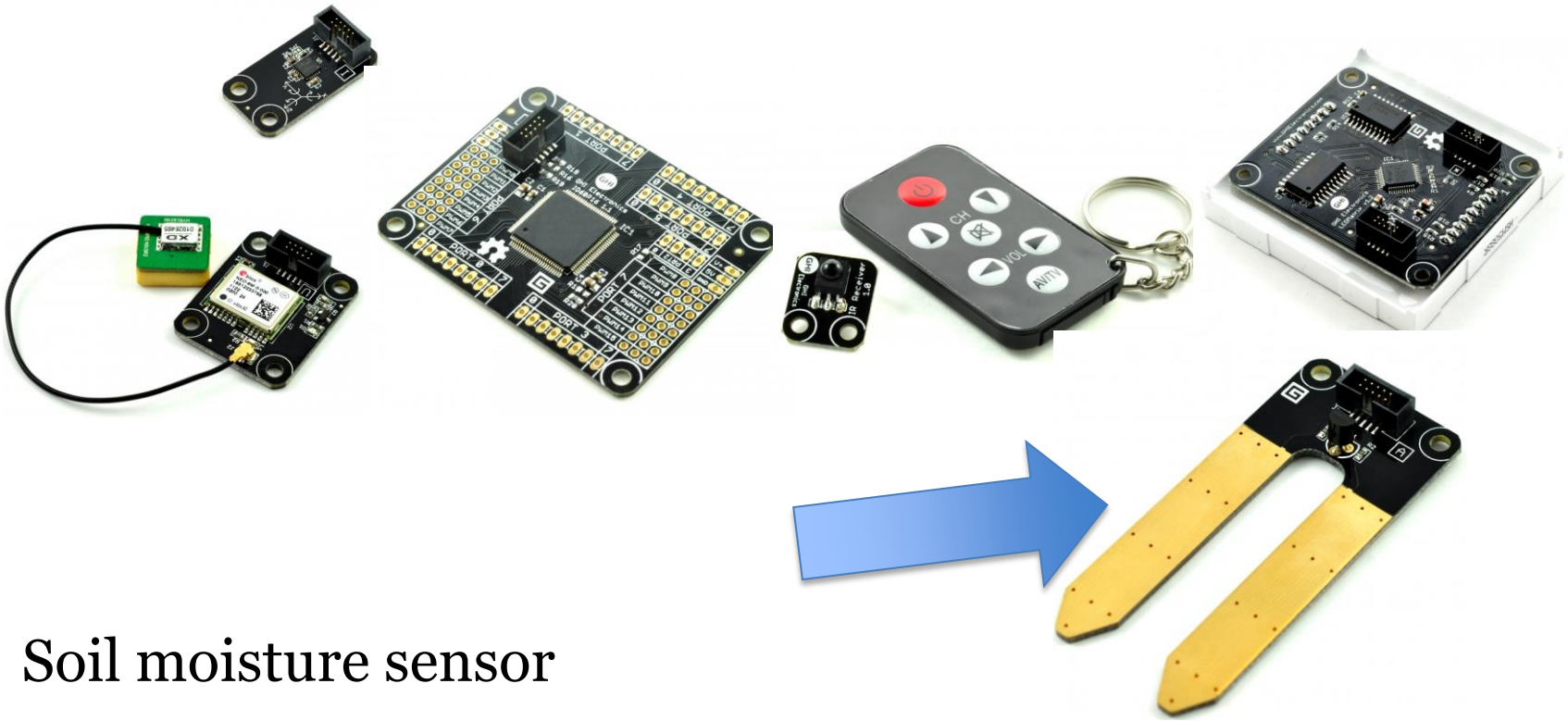
- Remote Control
  - Your gadget can be remote controlled

What do we mean by device?



- Light Matrix Control
  - Your gadget can drive 64 multi-coloured and ultra-bright LEDs

What do we mean by device?



- Soil moisture sensor
  - Your gadget can water your plants

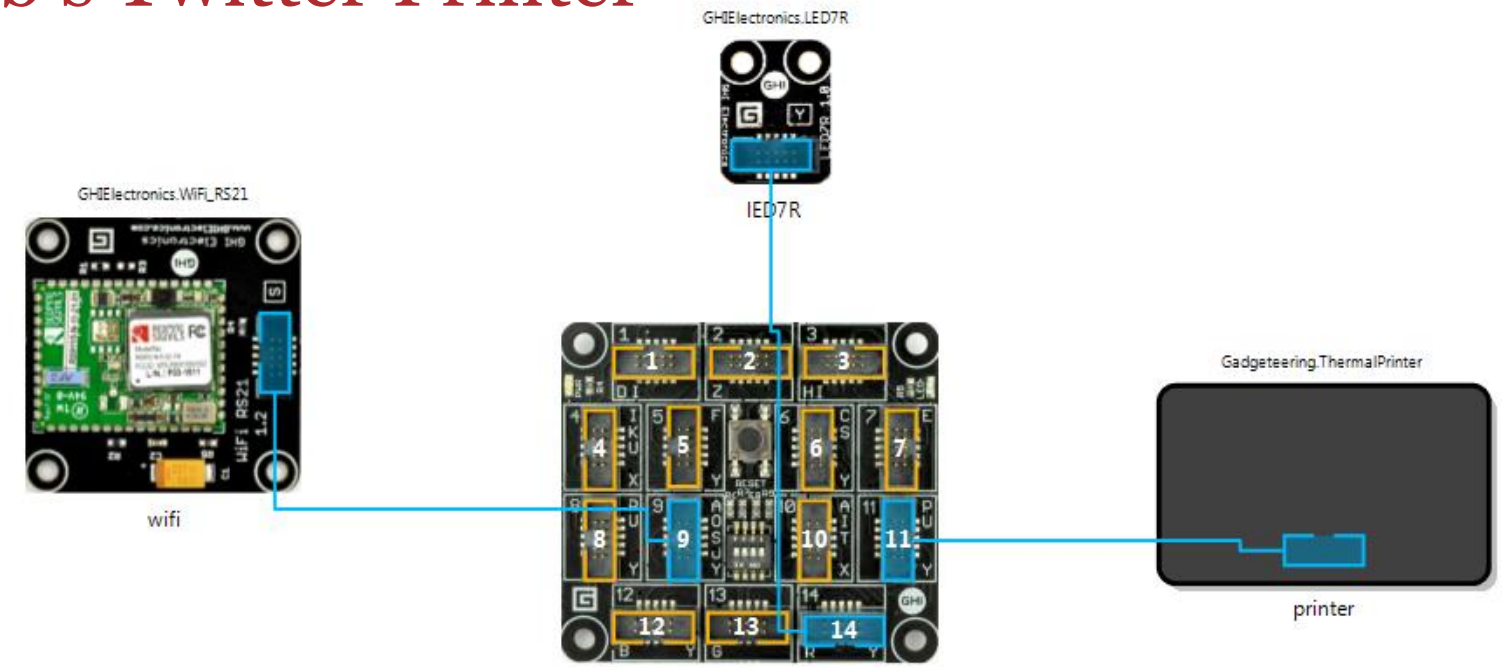
I could go on…

- There are also devices which can
  - Measure your heartbeat
  - Detect intruders
  - Measure temperature
  - Sense Light
  - Output video
  - Play Music
- These all connect via the sockets that you see on the processor board

Making Gadgets is easy

- Each of the devices connects to the computer in the same way, and we have a "software object" that represents the device
- We write programs that interact with the object
- There are basically two kinds of program that we make
  - Ones that respond when the user does something (take a picture when the user presses a button)
  - Ones that perform a task regularly (read the air pressure and then send it to a server on the network)

Rob's Twitter Printer



- I thought it might be fun to create a program that prints my tweets off Twitter
- I can use a Gadgeteer device to do this, arranged as you see above

demo

“Printing Tweets”

Demo 2: A Twitter Printer

Summary

- Computer controlled gadgets are easy and fun to make
- There is a huge range of different devices you can connect to
- A program can run in response to a stimulus, or at regular intervals
- You can now use 3D printers to produce the cases for your devices

Useful Stuff

- Gadgeteer Hardware and Software
  - http://www.netmf.com/gadgeteer/
  - http://www.ghielectronics.com/
- Free 3D Design Tools
  - http://sourceforge.net/projects/free-cad/
  - http://www.123dapp.com/
- 3D Printers
  - http://www.ultimaker.com/
- My Blog
  - http://www.robmiles.com