오픈소스와 임베디드시스템

2014.12-13
윤형기
(142037@sdu.ac.kr)
순서

• 도입
• 오픈소스
  – 오픈소스 소프트웨어 (OSS)
  – 오픈소스 하드웨어 (OSHW)
  – SDx (Software-Defined Anything)
• 오픈소스 임베디드시스템
  – 플랫폼: Arduino, Rasberry Pi, Beagle B’d, Galileo, etc.
  – H/W와 S/W환경
  – 활용사례
• 오픈소스와 IOT
  – 오픈소스 IOT 플랫폼
  – 활용 사례
• 맺음말
도입
세상 변하는 얘기 (1)
세상 변하는 얘기(2)
Why Software Is Eating The World

By MARC ANDREessen
August 20, 2011

This week, Hewlett-Packard (HPQ) announced plans to jettison its struggling PC business, an area where it sees better potential for growth than mobile or cloud computing. Meanwhile, cellphone handset maker Motorola Mobility (MOT), which Google (GOOG) acquired for $12.5 billion, is also in line with Android, the company's new mobile operating system. Both moves are also in line with the company's strategy to focus on growth and innovation in the stock market.

Summary: If we're to build a software-powered world, what should we be thinking about? Why APIs are eating the world
오픈소스
오픈소스

Apache Hadoop Ecosystem

Management & Monitoring
(Ambari)

Scripting
(Pig)

Machine Learning
(Mahout)

Query
(Hive)

NoSQL Database
(HBase)

Distributed Processing
(MapReduce)

Workflow & Scheduling
(Oozie)

Distributed Storage
(HDFS)

Data Integration
(sqoop/REST/ODBC)

Linux

The Apache
Software Foundation
http://www.apache.org/
오픈소스 = ...

- 오픈소스란?
  - = Open Recipe
  - = 공유
  - = 커뮤니티
OSS의 역사

• 1960's ARPANET, ...
• 1969 Unix
• 1980 Usenet
• 1983 GNU 프로젝트
• 1985 FSF
• 1989 386BSD, FreeBSD, ...
• 1991 Linux kernel
• 1994 MySQL
• 1996 Apache 웹 서버
• 2001 Open Source 선언:
• 2004 Ubuntu

프로그래밍 언어의 역사

• Before C:
  – 1957 FORTRAN/ 1959 COBOL/ 1964 BASIC
• C
  – 1969 C
  – 1973 PASCAL
• C++
  – 1983 C++
    • http://www.youtube.com/watch?v=JoVQTPbD6UY
• After C/C++
  – 1991 Python
  – 1995 Java, Javascript
  – 1995 R
• OSI가 제시하는 OSS의 조건
  – (1) 자유배포
  – (2) 소스코드 공개
  – (3) 2차적 저작물 (Derived works) 허용
  – (4) 소스코드 수정 제한
  – (5) 개인이나 단체에 대한 차별 금지
  – (6) 사용분야에 대한 제한금지
  – (7) 라이선스의 배포
  – (8) 라이선스 적용상의 동일성 유지
    • (라이선스 must not be specific to a product)
  – (9) 다른 라이선스의 포괄적 수용
    • (라이선스 must not contaminate other software)
  – (10) 라이선스의 기술적 중립성
    • (라이선스 must be Technology-Neutral)
• 주요 라이선스

출처: https://www.blackducksoftware.com/resources/data/

• http://opensource.org/licenses
개방형 표준

• 개방 표준 (Open Standards)의 요건
  – No Intentional Secrets:
  – Availability:
  – Patents:
  – No Agreements:
  – No OSR-Incompatible Dependencies

• 누구나 구현 가능
오픈소스 하드웨어

• 개념
  - 좁은 의미: www.oshw.org
  - 넓은 의미: …

• 제공되어야 할 것
  - 원본 디자인 파일(Original Design Files)
  - 보조 디자인 파일(Auxiliary Design Files)
  - 재료 명세서(Bill Of Material)
  - 소프트웨어 및 펌웨어
  - 사진, 기타 설명서(Instructions)

• 배경
  - 1970년대
  - 1980~90년대 - OSS 운동
  - 1990년대 중반 이후 - OSHW 확산: SOC, FPGA, 임베디드시스템
<table>
<thead>
<tr>
<th>구 분</th>
<th>내용</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSHW의 원칙</td>
<td>OSHW는 누구든지 특정 디자인이나 해당 디자인에 근거한 HW를 학습, 수정, 배포, 제조, 판매할 수 있도록 공개된 HW이다. HW를 만들기 위한 디자인 소스는 수정하기에 적합한 형태로 제공되어야 한다. OSHW는 각 개인들이 HW를 만들고 그 사용을 극대화할 수 있도록 쉽게 구할 수 있는 부품과 재료, 표준 가공 방법, 개발된 시설, 제약 없는 콘텐츠, 오픈소스 디자인 툴 등 을 사용하는 것이 이상적이다. OSHW는 디자인을 자유롭게 교환함으로써 지식을 공유하고 사용화를 장려함으로써 사람들로 하여금 자유롭게 기술을 제어할 수 있도록 한다.</td>
</tr>
<tr>
<td>OSHW의 정의</td>
<td>OSHW는 누구든지 제작, 수정, 배포하고 사용할 수 있도록 디자인이 공개되는 물리적 인공물 (기계, 장비 및 기타 실제가 있는 물건)을 나타내는 용어이다. 본 정의는 OSHW의 라이선스 개발 및 평가를 위한 지침을 제공하는데 도움이 되는 것을 목적으로 하고 있다. 물리적 제품을 만드기 위해서는 물리적인 자원의 투입이 반드시 필요하다는 점에서 HW는 SW와 상이하다. 따라서 아이템(제품)을 생산하는 개인이나 회사는 OSHW 라이선스에 따라 생산한 제품이 본래의 디자이너에 의해서 제작, 판매, 보증 또는 승인되지 않았음을 명시하고, 본래 디자이너의 상표를 사용하지 않아야 할 의무가 있다.</td>
</tr>
</tbody>
</table>
# OSHW 배포 조건

<table>
<thead>
<tr>
<th>문서</th>
<th>OSHW는 디자인 파일을 포함한 문서와 함께 공개되어 있어야 하며, 해당 디자인 파일은 수정 및 배포 가능하다. 문서가 실제 제품에 포함되어 있지 않은 경우에는 인터넷 무료 다운로드와 같이 잘 알려진 방법을 통해 문서를 제공함으로써 재생산 비용이 합리적인 수준을 초과하지 않도록 한다. 문서는 CAD 프로그램의 원본 파일과 같이 수정에 적합한 형식의 디자인 파일을 포함해야 한다. CAD 프로그램에서 생성된 코퍼 그림(Copper artwork)의 인쇄 데이터와 같이 컴파일된 컴퓨터 프로그램과 유사한 중간 형태로는 대체할 수 없다. 라이선스는 디자인 파일이 완전히 문서화된 오픈 파일 형식을 요구할 수도 있다.</th>
</tr>
</thead>
<tbody>
<tr>
<td>문서</td>
<td>OSHW의 문서는 그 전체가 공개되지 않는 것이라면 라이선스 하에 공개된 것이 어느 부분인지 정확하게 명시해야 한다.</td>
</tr>
</tbody>
</table>
| 필요한 SW | 라이선스 디자인이 제대로 작동되고 필수 기능을 충족시키기 위해 임베디드 또는 다른 형태의 소프트웨어를 필요로 하는 경우에는 라이선스는 다음의 조건 중 하나를 충족시킬 것을 요구할 수 있다.

a) OSS를 쉽게 작성해 기기가 정상적으로 작동하고 필수 기능을 충족시킬 수 있도록 인터페이스에 대한 문서화가 충분해야 한다. 즉, 문서에서 자세한 신호의 타이밍 다이어그램 또는 작동 과정에서 인터페이스를 정확하게 설명하는 의사 코드(pseudocode)에 대한 사용법 등이 포함될 수 있다.

b) 필요한 SW는 ‘오픈소스이니셔티브 (Open Source Initiative, OSI)’가 승인한 오픋소스 라이선스 하에 배포된다. |
<table>
<thead>
<tr>
<th>내용</th>
<th>설명</th>
</tr>
</thead>
<tbody>
<tr>
<td>파생물</td>
<td>라이선스는 변경과 파생물을 허용하며 원본과 동일한 라이선스 하에서 배포되는 것을 허용한다. 라이선스는 제조, 판매, 배포, 디자인 파일로부터 만들어진 제품의 활용, 디자인 파일 자체 및 그 파생 작업을 허용한다.</td>
</tr>
<tr>
<td>자유로운 재배포</td>
<td>라이선스는 어떠한 단체에 대해서도 프로젝트 문서의 판매 및 배포를 제한해서는 안된다. 라이선스는 이러한 판매에 대한 사용료 및 로열티를 요구해서는 안되며, 파생물의 판매에 대해서도 사용료나 라이선스를 요구해서는 안된다.</td>
</tr>
<tr>
<td>귀속</td>
<td>라이선스는 디자인 파일, 생산된 제품, 파생물 자체를 유통할 때 라이선스 보유 주체에게 귀속 권한을 주기 위해 파생된 문서와 기기 관련 저작권 표시를 요구할 있다. 라이선스는 제품 또는 기기 소비자(end-user)가 이러한 정보에 접근할 수 있도록 하고 있으며, 정보의 표식 형식에 제한은 없다. 라이선스는 파생제품에 대해서 본래의 디자인과는 다른 제품 번호나 제품명을 사용하도록 요구할 수 있다.</td>
</tr>
<tr>
<td>개인이나 단체의 차별 금지</td>
<td>라이선스는 어떠한 개인이나 단체에 대해서도 차별 없이 적용된다.</td>
</tr>
<tr>
<td>활동 분야에 대한 차별 금지</td>
<td>라이선스는 제작물의 특정 활동 분야에서의 이용을 제한해서는 안된다. 예를 들어, OSHW가 일반 사업이나 핵 연구에 사용되는 것을 제한해서는 안된다.</td>
</tr>
<tr>
<td>라이선스 배포</td>
<td>라이선스에 의해 승인된 권리는 추가적인 라이선스 집행 없이도 재배포 된 모든 제작물에 적용된다.</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>라이선스의 특정제품 국한 금지</td>
<td>라이선스에 의해 승인된 권한은 특정 제품의 일부에 포함된 라이선스 제작물에만 국한되지 않는다. 만일 제작물에서 일부분이 추출되어 해당 라이선스 하에서 활용 또는 배포된 경우 그 제작물을 배포한 각 당사자는 원래 제작물에 주어진 것과 같은 권한을 갖는다.</td>
</tr>
<tr>
<td>타 HW 및 SW 제한 금지</td>
<td>라이선스는 라이선스된 제작물의 통합이나 파생에 제한을 두지 않는다. 예를 들어 라이선스는 라이선스된 제작물과 함께 판매되는 HW를 모두 오픈소스화해서 판매하거나 기기 내부에 OSS만 사용하도록 강제하지 않는다.</td>
</tr>
<tr>
<td>라이선스의 기술 중립성</td>
<td>라이선스는 그 어떤 항목도 개별 기술, 특정 부분 · 부품 · 재료 · 인터페이스 스타일 · 활용에 국한되거나 제한하지 않는다.</td>
</tr>
</tbody>
</table>
### 오픈소스 하드웨어 라이선스

<table>
<thead>
<tr>
<th>라이선스</th>
<th>MIT 라이선스</th>
<th>Simplified BSD 라이선스</th>
<th>Modified BSD 라이선스</th>
<th>Creative Commons - Attribution 3.0</th>
<th>Creative Commons - Attribution - Share Alike 3.0</th>
<th>TAPR Open Hardware 라이선스 (OHL)</th>
<th>GPL/LGPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>원 저작자 명시</td>
<td>Optional</td>
<td>Optional</td>
<td>NO</td>
<td>Required</td>
<td>Required</td>
<td>Optional</td>
<td>NO</td>
</tr>
<tr>
<td>2차 저작물도 오픈소스여야 하나?</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>2차 저작물은 다른 라이선스를 택할 수 있나?</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>2차 저작물의 판매가능성?</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Patent restrictions</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

출처: [http://www.inmojo.com/licenses/](http://www.inmojo.com/licenses/)
• The value of disaggregation!
  – 기존제품을 핵심부품별로 분해 (break down)하고 재설계
  – 업계간 협업을 통한 OS-agnostic top-of-rack (TOR) switches.
• 유연하면서도 경쟁상태를 유지
SDx (S/W-Defined Anything)

- SDx – Umbrella term
  - SDN: s/w-defined networking
  - SDC: s/w-defined computing
  - SDDC: s/w-defined datacenter
  - SDS: s/w-defined storage and SN.

- 컴퓨팅 인프라를 가상화하고 “서비스”로 제공
  - 모든 컴퓨팅 자원의 관리와 제어가 intelligent s/w에 의해 자동화

- 의미
  - 공급사 중심의 black box → Commodity 즉, ownership의 변화
  - Vendor 독점 두뇌 → 협력하는 두뇌
  - 망의 변화를 통해 신규 서비스 제공이 보다 유연해짐

Software Defined ==
API Enabled

- Programmable Everything

출처: Gartner
SDx 배경: SDR

- 시조: Alexander Graham Bell
  - 1876 전화기 발명
- Nyquist와 Shannon
  - Sampling/ Modulation (ADC/DAC)
  - Signal 이론
- SDR, Cognitive Radio
  - Programmable Digital H/W
  - Programmable Analog H/W
- Fourier works in digital, too: “Discrete Fourier Transform”
New Paradigm: SDR

- 원리
  - RX 신호를 LNA 직후 S/W로 digitize, S/W로 처리
    - ➔ flexible TX, vice versa

- Programmmable SDR: 현황과 전망
  - “Classical” way:
    - VERILOG
    - 문제: H/W (board) specific, rewrite for different boards
  - State-of-the-art
    - SCA (S/W Comm. Architecture)
    - CORBA, XML with ORB middleware
  - 전망
    - Ontology 기반 - “describe” to the board what to do, board (compiler) figures out how
  - 장점
    - Board 프로그래밍이 용이해지고 architecture별 재작업이 감소
    - reconfguration “on-the-fly”
    - 신규 정책 적용이 용이
• Open Source SDR
  – HPSDR
    • GNU와 유사한 "next generation" SDR
    • Radio Amateurs ("hams") 및 Short Wave Listeners (SWLs)
  
  – GNURadio
    • signal processing blocks to implement S/W radios
    • GPL v3

  – Universal Software Radio Peripheral (USRP)
  – ...
SDN

현재 데이터센터의 문제점

- 네트워크 장비
  - 솔루션들의 단순 나열
    - 다음 항목에 대한 대처가 부족
      - mobile device 및 access 관리
      - mobile security 및 DDOS
      - Application layer threats
  - 운영 불일치
    - 업무 및 운영 policy에서의 Inconsistency
    - Application 성능 및 보안의 불안

- 서비스 아키텍처
  - 각 서비스가 단절되어 있음
    - 별도의 보안, load balancing, authentication/authorization, Layer 7 서비스
    - More applications need services

- 비호환 API의 이용
  - Proxy를 이용하지만
    - API versioning, Client-based steering
    - API Load Balancing, Metering & Billing
    - API key management, ...

2014-12-13
• SDN의 특징
  – control plane과 data plane 분리
  – S/W로 제어 (API 지원)
  – 네트워크 지능을 가상화
  – ONF (Open Network Foundation)

  • Configuration Interfaces
  • CLI, GUI, REST API
  • Policies
    • Routing FW/ NAT
    • Policy QoS
    • VPN Termination Network
    • Bridging Interface Drivers
시스코, 아마존 프로젝트에 놀라 SDN 올인 결정

임만철 기자/ imc@zdsnet.co.kr 2013.11.01 / PM 02:17 시스코, 인시스애미, SDN

아마존에 10억달러급 규모의 네트워크 장비 공급 계약을 맺으려 했지만 뜻대로 되지 않았다. 아마존은 겨우 1천100만달러치 장비만 사들였다. 아마존은 시스코의 고가 장비 대신 상대적으로 저렴한 하드웨어(HW)와 SDN 기술로 충당

그리고 챌버스 CEO는 고위 임원들에게 만일 시스코가 SDN 시장에 진출할 경우 어떤 일이 벌어질 것인지 분석할 것을 지시했다. 이 경우 시스코 사업은 430억달러 규모에서 220억달러로 줄어들 것이라는 결론이 나왔다.
- Google router가 전세계 router의 7%
  - SDN + OpenFlow: Bandwidth utilization = 40% → 97%
• SDN Applications
  – Programmable Open API를 이용한 Cloud Orchestration

• SDN Control Plane
  – Controller
  – NOS

• SDN Data Plane
  – Network Devices
  – OpenFlow
OpenStack

- S/W 업체가 참여할 수 있는 일종의 kernel을 제공코자 함.
- OpenStack 프로젝트
  - Compute: Nova, Glance
  - Storage: Swift, Cinder
  - Networks: Quantum
  - Tools: Horizon, Ceilometer, Heat
  - Security: Keystone

OpenFlow

- SDN의 핵심: OpenFlow
  - 스위치의 작업을 제어 (dynamically/ programmatically)
  - Flow tables
  - 하위 프로젝트
    - NOX, POX, Beacon, Floodlight
  - OpenDaylight
    - Linux Foundation의 프로젝트
활용 사례
- Amazon
  - EC2, S3, EMR, ...
- Rackspace
  - Compute: Cloud servers
    - 가상화된 API-accessible servers
    - Xen & XenServer HVs
  - Storage: Cloud Files
    - Object file store
  - PaaS: Cloud Sites

예: F5 Synthesis
- Software Defined Application Services (SDAS)
  - 36개 장비로 하나의 시스템 구성
  - Elastic, multi-tenant platform
  - Application-aware, BigIQ
- 고성능 Services Fabric
  - 20.5 TByte의 combined throughput
  - 92억 개의 connection capacity (20억명 동시 session)

F5 Synthesis Partner Ecosystem
• Software-Defined Storage

- RAID의 개념에서 발전 (1988)
SDDC (Software-Defined Datacenter)

• 개념

• Cloud-ready Datacenter
  – 요건
    • Homogeneous configuration
    • 전력밀도의 확장
    • Fat Tree/Mesh networks
  – Zettabyte
    • = 1,000 Exabytes = 1,000,000 Petabytes
    • 현재 지구상 데이터 총량 (150GB/person), 2% on earth in 2020
    • Data must be stored efficiently
  – = 세계 GDP의 38%!
• **Leave No Service Behind!**
  – Leave no applications behind!

• 미래의 Cloud
  – 개념
    • 서비스 제공자의 cloud 환경을 연결하고 load balance
  – Cloud Federation

  – SAML
    • Internet SSO
    • Security
    • Access
    • Administration
오픈소스 임베디드 시스템
오픈소스 임베디드 시스템?

• DIY
  - 직접 조립
  - 직접 프로그래밍
• 오픈소스
  - 오픈소스 소프트웨어
  - 오픈소스 하드웨어
• 임베디드 시스템
  - H/W + S/W + F/W
  - 특정용도 특화
  - Sensor + <Microcontroller OR Microprocessor> + Actuator
• Prototyping 플랫폼
  - 모형제작 OR 취미
  - (제한적으로) Field Deploy
  - 향후: ??
<table>
<thead>
<tr>
<th></th>
<th>Arduino Uno</th>
<th>Raspberry Pi B+</th>
<th>Beaglebone Black</th>
</tr>
</thead>
<tbody>
<tr>
<td>제작사 및 배경</td>
<td>Arduino (이탈리아) IDI (대학)</td>
<td>Raspberry (영국) Cambridge 대학</td>
<td>Beagleboard (미국) Texas Instrument</td>
</tr>
<tr>
<td>CPU</td>
<td>ATmega 328 16 MHz</td>
<td>ARM 700MHz</td>
<td>ARM Cortex-A8 1GHz</td>
</tr>
<tr>
<td>RAM</td>
<td>2KB</td>
<td>512MB</td>
<td>512MB</td>
</tr>
<tr>
<td>Flash Memory</td>
<td>32KB</td>
<td>-</td>
<td>2GB eMMC</td>
</tr>
<tr>
<td>OS</td>
<td>Firmware (toolkit)</td>
<td>Linux, Android</td>
<td>Linux, Android</td>
</tr>
<tr>
<td>Network</td>
<td>-</td>
<td>Ethernet</td>
<td>Ethernet</td>
</tr>
<tr>
<td>주된 사용용도</td>
<td>Sensing/Actuator</td>
<td>SBC +</td>
<td>SBC +</td>
</tr>
<tr>
<td>가격</td>
<td>$35</td>
<td>$35</td>
<td>$45</td>
</tr>
<tr>
<td>S/W 개발도구</td>
<td>Ard IDE, Processing</td>
<td>Scratch, Python</td>
<td>Cloud9, Node.js</td>
</tr>
</tbody>
</table>

2014-12-13
<table>
<thead>
<tr>
<th></th>
<th>Arduino Uno</th>
<th>Raspberry Pi</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Processor</strong></td>
<td>AVR ATmega328p</td>
<td>Broadcom ARM1176JZF-S</td>
</tr>
<tr>
<td><strong>Clock Speed</strong></td>
<td>16MHz</td>
<td>700MHz</td>
</tr>
<tr>
<td><strong>Register Width</strong></td>
<td>8-bit</td>
<td>32-bit</td>
</tr>
<tr>
<td><strong>RAM</strong></td>
<td>2k</td>
<td>512 MB</td>
</tr>
<tr>
<td><strong>GPIO</strong></td>
<td>20</td>
<td>8</td>
</tr>
<tr>
<td><strong>I/O Current Max</strong></td>
<td>40mA</td>
<td>5-10 mA</td>
</tr>
<tr>
<td><strong>Power</strong></td>
<td>175 mW</td>
<td>700 mW</td>
</tr>
<tr>
<td><strong>Operating System</strong></td>
<td>None</td>
<td>Linux &amp; Others</td>
</tr>
</tbody>
</table>
• 종류:
  - ...

Arduino

RAM: 2K
Flash: 32K
Timers
Serial (UART)
I²C
SPI
Raspberry Pi

- SBC (Single Board Computer)
  - mainly with Linux

- Model B+
  - BCM2835 application processor
  - 512MB RAM.
  - 40 pin GPIO
  - 4 USB 2.0 ports, micro SD version.
  - 저전력, replacing linear regulators with switching ones
Beaglebone Black

- **Technology Specifications**
  - AM335x 1GHz ARM Cortex-A8
  - 512MB DDR3 RAM
  - 4GB 8-bit eMMC on-board flash storage
  - 3D graphics accelerator
  - 2x PRU 32-bit microcontrollers

- **Connectivity**
  - USB client for power & communications
  - Ethernet, HDMI, 2x 46 pin headers

- **Software Applications**
  - Debian, Android, Ubuntu
  - Cloud9 IDE on Node.js w/ BoneScript library
• 기술사양
  – Allwinner A20 ARM Cortex ARM7 dual-core processor.
  – 26-pin GPIO headers (Raspberry Pi와 호환)
  – OS: Linux and Android
  – $29.99
• 특징
  – On-board: 마이크, SATA 커넥터, IR 수신기, OTG connector, 전원 & reset 스위치
Intel Galileo Gen 2

- 기술사양
  - Intel Quark SoC X1000 & Pentium instruction set.
  - 표준 I/O 인터페이스(PCI, Ethernet, USB)
  - £60

- 특징
  - Certified as an Arduino open source development board,
  - Arduino IDE 이용 가능. (Mac OS, Windows, Linux)
  - Galileo Gen 2 itself runs Linux.
Intel NUC

• “Next Unit of Computing”
• 기술사항
  – Intel Core i3 3127U (dual-core 64-bit 1.8GHz)
  – 4-by-4-inch form factor
  – OS: Windows, Linux, Android, Chrome OS.
  – £100 ~ £300
• 특징
  – 고성능 compact PC alternative to Raspberry Pi.
  – Home theatres, digital signage, kiosks
  – Gateway to IoT
Intel Edison

- 기술사양
  - SD card 크기
  - dual-core Intel Quark x86 400MHz
  - 1GB RAM, 4GB Flash memory,
  - UART, GPIO, USB, Bluetooth, Wifi 포함
  - 70 pin dense connector
  - Python, Node.js 사용가능
  - 2014 발표
- Starter Kit (SparkFun)
<table>
<thead>
<tr>
<th></th>
<th>Arduino Uno</th>
<th>Raspberry Pi Model B+</th>
<th>Intel Edison</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Price</strong></td>
<td>$30</td>
<td>$35</td>
<td>$50 (board not included)</td>
</tr>
<tr>
<td><strong>Size</strong></td>
<td>7.6 x 1.9 x 6.4 cm</td>
<td>8.5 x 5.6 x 1.7 cm</td>
<td>3.55 x 2.5 x .39 cm</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>0.002MB</td>
<td>512MB</td>
<td>1 GB</td>
</tr>
<tr>
<td><strong>GPIO</strong></td>
<td>14</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td><strong>Clock Speed</strong></td>
<td>16 MHz</td>
<td>700 MHz</td>
<td>500 MHz, 100 MHz</td>
</tr>
<tr>
<td><strong>On Board Network</strong></td>
<td>None</td>
<td>10/100 BaseT Ethernet socket</td>
<td>Dual-band (2.4 and 5 GHz) Wifi, Bluetooth 4.0</td>
</tr>
<tr>
<td><strong>Multitasking</strong></td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Input voltage</strong></td>
<td>7 to 12 V</td>
<td>5 V</td>
<td>3.3 to 4.5 V</td>
</tr>
<tr>
<td><strong>Flash memory</strong></td>
<td>32KB</td>
<td>Micro SD card</td>
<td>4 GB eMMC</td>
</tr>
<tr>
<td><strong>USB</strong></td>
<td>One, input only</td>
<td>Four, peripherals OK</td>
<td>One, peripherals OK</td>
</tr>
<tr>
<td><strong>Operating System</strong></td>
<td>None</td>
<td>Linux distributions</td>
<td>Yocto Linux v1.6</td>
</tr>
<tr>
<td><strong>Integrated Development Environment</strong></td>
<td>Arduino IDE</td>
<td>Scratch, IDLE, anything with Linux support</td>
<td>Arduino IDE, Eclipse, Intel XDK</td>
</tr>
</tbody>
</table>

**H/W 환경**

<table>
<thead>
<tr>
<th>마이크로 프로세서</th>
<th>마이크로 컨트롤러</th>
</tr>
</thead>
<tbody>
<tr>
<td>컴퓨터의 CPU에 해당 - 주로 범용 시스템. 고성능</td>
<td>주로 embedded system에 이용 상대적으로 저성능</td>
</tr>
<tr>
<td>Only 처리장치. :: 메모리, (I/O)장치 필요 → 회로 크기, 비용, 전력소모 ↑</td>
<td>내부에 메모리와 I/O 장치 :: 회로크기가 작아지고 저 전력. 내부통신 → 빠른 처리속도</td>
</tr>
<tr>
<td>적은 수의 register → 주로 메모리 작업</td>
<td>많은 수의 register → 프로그래밍이 용이</td>
</tr>
</tbody>
</table>

Schematic Symbols for Common Electronics and Electrical Components
## 소프트웨어 환경 비교

<table>
<thead>
<tr>
<th></th>
<th>Arduino</th>
<th>Raspberry Pi</th>
<th>Beaglebone Black</th>
<th>Intel Galileo</th>
</tr>
</thead>
<tbody>
<tr>
<td>운영체제</td>
<td>Toolkit</td>
<td>Pidora, Archlinux, Raspian (Debian); Android OS, Firefox OS, ...</td>
<td>Linux (Debian, Ubuntu, Fedora), BSD, Windows Embedded, 기타</td>
<td>Linux (Yocto) (with Grub)</td>
</tr>
<tr>
<td>개발환경</td>
<td>독자 IDE</td>
<td>Eclipse 등</td>
<td>Eclipse 등</td>
<td>Arduino IDE</td>
</tr>
<tr>
<td>프로그래밍 언어</td>
<td>Arduino C</td>
<td>Python 중심</td>
<td>Node.js 중심</td>
<td>Arduino C 또는 Linux 개발환경</td>
</tr>
<tr>
<td>라이브러리</td>
<td>Arduino Library</td>
<td>Linux 표준 library 등</td>
<td>Linux 표준 library 등</td>
<td>Linux 표준 library 등</td>
</tr>
<tr>
<td>기타</td>
<td>H/W 중심</td>
<td>SBC</td>
<td>SBC</td>
<td>Arduino with Intel Quark Inside! (400MHz)</td>
</tr>
</tbody>
</table>
• 임베디드 프로그래밍 언어
  - (Assembly)
  - C/C++
  - Python
  - Javascript/Node.js
  - Java (Java ME), 기타

• Bootloader
  - Uboot, GRUB, ... 및 제품별 수정

• 각종라이브러리
활용

• ArduCopter

  • http://www.youtube.com/watch?v=X4GkAmEUA9M
  • https://code.google.com/p/arducopter/

• BotaniCalls
  - 식물용 센서
  - 토양습도를 측정해서 식물에 물 줄 시점을 Twitter로 알림
  - 습도센서 + Arduino + 무선 Ethernet shield
  - 제작kit 판매: $99.95
  - http://www.youtube.com/watch?v=af6cayzWW1Y
• Arduboy
  – Arduino 이용 각종 게임기
  – Tetris 게임기

• RepRap프로젝트
  – Arduino (Sanguino라는 Arduino 파생보드) 기반의 오픈소스 3D 프린터
• **IMAGE-PROCESSING ROBOT**
  – Arduino, Raspberry Pi, ... +
  – OpenCV for Image processing

• **GPS-synchronised clock**
  – Arduino 이용
  – 정확한 UTC time by GPS-synchronized clock
  – external GPS 안테나와 태양광 (5 volts) 이용
• Rapiro 3D 프린터

• Node.js로 Arduino 제어
  – https://github.com/jinniahn/duino

DUINO 아두이노 프로그램 설치

USB Cable

DUINO Module + Node.js App

$> git clone https://github.com/jinniahn/duino
$> cd duino/src/du
$> openduino.ino
• Data Logging
  - Python을 이용하여 Arduino에서 데이터를 MySQL 데이터베이스에 logging
• Wall-mounted calendar
  – Raspberry Pi + HDMI-capable 모니터
  – dynamically updating Google calendar display
  – Web interface 지원

• PiPhone
  – Raspberry Pi 기반의 스마트폰
  – Adafruit touchscreen interface + Sim900 GSM/GPRS module
• Asterisk VoIP
  - Raspberry Pi 이용한 SOHO 용 Asterisk VoIP PBX VoIP 응용.
  - PBX connected to a GSM-to-SIP gateway (8 회선) gateway

• 온도 습도 모니터
  - Raspberry Pi
  - 여러 가지 센서로부터의 데이터는 MySQL 데이터베이스에 logging.
  - Web 인터페이스로 그래픽 처리.
• Web-controlled power strip
  – intelligent PDUs (power distribution units)
  – Web-controllable power strip

• Surveillance camera
  – Raspberry Pi
  – 단거리에서는 passive POE injector (10/100 Ethernet only!) 사용 가능 (single cable).
  – motion-detection 및 recording 은 NAS 장비 이용
• RaspbAIRy
  – Raspberry Pi AirPlay speaker
  – shairport Linux 기반
  – amplifiers, speakers 등을 통합 steampunk radio.

• Supercomputer Raspberry Pi
  – 64 개의 Raspberry Pi 를 Ethernet switch로 연결
  – → 1 TB memory (care of 64 different 16GB SD cards)
• **FishPi**
  – Drone boat를 대서양 횡단, 과학데이터 측정 계획
  – 크기: 20” from stem to stern, the vessel is powered by a 130-watt solar panel.

• **Indoor weather station**
  – Raspberry Pi
  – 실내 작업환경 측정 후 cloud로 데이터 전송처리
  – 측정대상: 온도, 습도, 밝기, 기압, noise pollution, …
• Beer can keyboard
  – Raspberry Pi 이용한 QWERTY 키보드
  – 44개의 맥주 캔 키보드를 두드리면 plasma 스크린에 출력.

• Teeny tiny arcade
  – Raspberry Pi
  – arcade-style gaming cabinet.
  – laser-cut plastic 상자 + 2.4-inch TFT display.
• Raspberry Pi board를 이용한 포터블 컴퓨터.
  – 자동차 거울을 모니터로 사용.
  – mini-wireless keyboard, a USB power pack, laser-cut transparent base.

• Solar-powered FTP 서버
  – Raspberry Pi + 태양광 패널 + compartments for the computer with holes in all the right places for I/O ports and four AA rechargeable batteries.
• Pi in the sky
  – GPS를 이용한 고공 풍선 (balloon)의 고도 측정 및 제어
  – GPS radio, webcam 및 각종 센서 활용.
  – 고공 촬영 데이터 등을 블로그 및 Flickr에 게재

• Picture-perfect Raspberry Pi
  – battery grip을 이용해서 DSLR 카메라에 컴퓨터 내장
  – 찍은 사진을 실시간으로 컴퓨터에 wifi 전송 및 원격 제어
• CNC hot-wire foam-cutting machine
  - 공작도구를 통한 제작환경 구축

• Palmtop Pi
  - Pi-to-Go minicomputer
  - 640x480 pixel 화면, touchpad, 64GB solid-state drive,
  - Wi-Fi, Bluetooth, HDMI 지원
IOT
IoT란?

• 인터넷에 연결된 식별가능한 사물(thing)이 인간의 개입 없이 서로 정보를 주고받으면서 인간 중심적 서비스를 제공할 수 있도록 해 주는 인프라 기술

![Human Beings vs, Internet Connected Devices (millions)]

Source: Cisco Systems, LM Ericsson, Raymond James research.

• 실제 현실에서...
  - 각 가정 인터넷 PC 1대 개인 1대 스마트폰 태블릿, 스마트 TV, 게임콘솔, 프린터
  - 1인 1대를 넘어섰음 (IPv6 필요성) 사물이 직접 인터넷에 연결

2014-12-13
IoT/IoE/M2M: “As a Service”

- **IoE/M2M**
  - Car as a Service
  - Home as a Service
  - Farm as a Service
  - Product as a Service
  - Healthcare as a Service
  - Factory as a Service
  - Food as a Service
• 출처: Cinterion, Presentation to 3rd FUSECO Forum, 2012
• 센서
  ✓ 실제 데이터를 수집하는 최적점

• 임베디드 단말
  ✓ 수집된 정보를 취합 및 전송

• 네트워크
  ✓ 근거리: 주로 센서와 임베디드 단말 통신
  ✓ 원거리: 단말과 서버 사이의 통신

• 미들웨어
  ✓ Event Processing: 대량 데이터에 대한 1차 패턴 분석
  ✓ Messaging: 데이터의 전송 보장

• 데이터 처리 및 분석
  ✓ 빅데이터: 센서에서 수집된 비정형, 대량 데이터 처리
  ✓ DW/BI: 데이터로부터 의미 있는 비즈니스 정보 추출

• 보안
  ✓ 데이터 유출, 외부 침입, 사생활 침해 등의 방지

• 클라우드 컴퓨팅
  ✓ 효율적인 End-to-End IoT 플랫폼 구축
오픈소스 IoT 도구

• 프레임워크
  – M2MLabs Mainspring
    • M2M application 개발용의 오픈소스 프레임워크
    • 주된 기능
      – 원격 monitoring, fleet management, smart grid 응용
      – Device modeling, configuration, device-application간 통신
      – 데이터 관리 (validation, normalization, retrieval, long-term storage)
    • 플랫폼: Java, Apache Cassandra NoSQL DBMS.
  – M2MLabs Mainspring
    • M2M application 개발용의 오픈소스 프레임워크.
• 개발도구
  – Eclipse IoT Project
    • Eclipse의 IoT 지원: MQTT CoAP, OMA-DM, OMA LWM2M + Lua (Eclipse의 IoT 언어).
    • 기타 Eclipse 관련 IOT 프로젝트: Mihini, Koneki, Paho.
  – Kinoma (3개의 subprojects)
    – Kimona Create = a DIY construction kit for prototyping
    – Kimona Studio = IDE
    – Kimona Connect = iOS and Android app that links smartphones and tables.
  – Node-RED
    • Node.js 기반의 "IoT 개발을 위한 visual tool"
    • Raspberry Pi 상에서 이용 가능
    • > 60,000 modules available to extend its capabilities.
- 홈 오토메이션
  - OpenHAB
    - Java-enabled hardware-neutral Home Automation
  - The Thing System
    - 가정 내에서 모든 Internet-connected things를 찾아냄
      - Nest thermostats, Samsung Smart Air Conditioners, Insteon LED Bulbs, etc.
    - 작성: Node.js
    - Raspberry Pi 이용 가능.
  - Freeboard
    - 사용자가 직접 자신의 dashboard를 작성해서 IoT deploy를 관리.
  - Exciting Printer
    - IoT printing - 소형 프린터로 IoT device에서 얻어진 정보 출력.
• **미들웨어**
  – IoTSyS
    • provides a communication stack for smart devices.
    • 지원표준: IPv6, oBIX, 6LoWPAN, Efficient XML Interchange.
  – OpenIoT
    • 오픈소스 cloud solution for IOT
    • Middleware for getting information from sensor clouds, without worrying what exact sensors are used."
    • 목표시스템: cloud 기반의 "sensing as a service,"
    • 다양한 성공사례 – smart agriculture, intelligent manufacturing, urban crowdsensing, smart living, smart campuses.
• 기타 – 플랫폼/통합도구
  – RIOT
  – Spark
  – DeviceHive
  – Devicehub.net
  – IoT Toolkit
  – Mango
  – Nimbits
  – OpenRemote
그 외의 오픈소스 프로젝트

• OpenHAB 프로젝트
  – http://www.openhab.org/
  – a vendor and technology agnostic open source automation software for your home.
• Tessel 프로젝트
  – https://projects.tessel.io/projects
• ThingSpeak 프로젝트
  – https://thingspeak.com/
• Hackster 프로젝트
  – Build internet-connected hardware.
  – http://www.hackster.io/spark
• BeagleBone Black 프로젝트
  – full featured, internet enabled 개발 플랫폼
  – low cost Sitara™ AM3358 ARM® Cortex™-A8 processor from TI 이용
  – 운영체제: Debian, Angstrom, Ubuntu 및 Android.
• Spark Core
  – https://www.spark.io/
• Project ara
  – Modular smartphones 제작을 위한 오픈 하드웨어 프로젝트
    • display, keyboard, extra battery 등을 마음대로 선택
    • Google 주도
  – 목표
    • 2015 년, BOM cost: $50 (for a basic grey phone)
  – 현재 MDK (Module Developer Kit) available
• Fab Lab
  – 레이저 커터, 3D 프린터 등 각종 디지털 장비를 사용해 개발자들이 시제품을 만들 수 있도록 도와주는 공작소
  – Fabrication + Laboratory
  – 미국) '쿼키(Quirky)', '테크숍(Techshop)' '이노센티브' 등

• Hackerspaces
  – http://hackerspaces.org/wiki/
  – = community-operated physical places, where people can meet and work on their projects.
  – ; to share their hackerspace stories and questions with the global hackerspaces community.

• Bug Labs
  – 오픈소스 H/W개발업체
  – Ford와 협력하여 OpenXc개발
  – 차량의 주행위치, 속도, 브레이크, 연료잔량 등의 내부 데이터를 실시간으로 다룰 수 있는 Arduino 및 Android
• 3D Robotics
  – 3D Robotics Iris Unboxing
  http://www.youtube.com/watch?v=4kTLDjx_jnY
  – IRIS+Drone - Flight modes
  http://www.youtube.com/watch?v=n2JtSQ7vDGA

• Spark
  – 자동온도조절기 = 온도센서 + Arduino + Wifi 모듈
  – Spark가 개발 ($70)
활용

- http://diydrones.com/
- http://aeroquad.com
- **Open Desk**
  - [https://www.opendesk.cc/](https://www.opendesk.cc/)
  - digital fabrication을 통해 designer와 maker를 직접 연결

- **Bio**
  - [http://lapaillasse.org/](http://lapaillasse.org/)
  - Open source bio-hacking
  - ink produced by bacteria by synthesizing a blue ink
• **Protei**
  – [https://sites.google.com/a/open sailing.net/protei/](https://sites.google.com/a/opensailing.net/protei/)
  – Open Hardware Oil Spill Cleaning Sailing Robot

• **OpenRov**
  – [http://community.openrov.com/%20](http://community.openrov.com/%20)
  – open source submarine rover
• Microfactory by 3D Printer
  – https://localmotors.com/
  – 3D-printed vehicle

• Mamba3D – 오픈소스 3D Printer
  – All-metal, 200 x 200 x 200 mm build area
 gratuites
개방과 참여

“20세기의 정치가 좌와 우의 대결이었다면
21세기에는 개방(Open)과 폐쇄(Closed)의 대결이다.”

– ALEC ROSS
Open Hardware 사업모델

• 예시
  – “Give away the bits, sell the atoms”
  – Charge 2.6x BOM
  – Keep ahead of cloners by innovating faster, supporting better
  – “90-10” Rule:
    • 90% the performance of commercial products at 10% the price
  – Democratize the technology:
    • low prices = high volume = high innovation.

• Source: Chris Anderson, MICROECONOMICS FOR MAKERS, Wired & 3D Robotics
전자, 영상처리 (opencv), 센서, robots, car, games, ...

나만의 컴퓨터, 내가 만든 컴퓨터
Thank You
Thank You
Thank You!!!